Treball final de grau

Desenvolupament d'una infraestructura *IoT* per a edificis intel·ligents

Grau en enginyeria informàtica

Especialitat d'enginyeria de computadors

Juny de 2015

Autor: Sergi Miquel Ribas

Director: Davide Careglio

Codirector: Sergio Ricciardi

Resum

L'*Internet* de les coses (*IoT*) és un concepte que cada vegada és més present en les nostres vides. Aplicar-lo en un edifici ens porta cap a la creació d'edificis intel·ligents mitjançant tecnologies informàtiques. En aquest treball es mostra com es poden fusionar aquests conceptes per crear una infraestructura que faciliti la creació d'aplicacions que ens permetin optimitzar els recursos energètics. L'objectiu és millorar el confort de les persones que fan ús d'un edifici públic i a la vegada millorar-ne l'eficiència energètica. Per fer-ho, es dissenya una xarxa de sensors i actuadors que permeten automatitzar el control de qualsevol sistema que consumeixi energia com la llum, la calefacció, l'aire condicionat, etc. Així doncs, es connecten els diferents espais d'un edifici a *Internet,* oferint dades sobre l'estat actual d'una zona concreta i permetent crear programes que automatitzin processos que finalment derivaran en actuacions per reduir el consum energètic, sense oblidar el confort de les persones.

Resumen

El *Internet* de las coses (*IoT*) es un concepto cada vez más presente en nuestras vidas. Aplicarlo en un edificio nos traslada a la creación de edificios inteligentes mediante el uso de tecnologías informáticas. En este trabajo se muestra cómo se pueden fusionar estos conceptos para crear una infraestructura que facilite la creación de aplicaciones que nos permitan optimizar los recursos energéticos. El objetivo es mejorar el confort de las personas que hacen uso de un edificio público y a la vez mejorar la eficiencia energética. Para hacerlo, se diseña una red de sensores y actuadores que permiten automatizar el control de cualquier sistema que consuma energía como la luz, la calefacción el aire acondicionado, etc. Así pues, se conectan los diferentes espacios de un edificio a *Internet,* ofreciendo datos sobre el estado actual de una zona concreta y permitiendo crear programas que automaticen procesos que finalmente derivaran en actuaciones para reducir el consumo energético, sin descuidar el confort de las personas.

Abstract

The *Internet* of things (*IoT*) is a concept every time more present in our lives. Apply that concept to a building takes us to the creation of smart buildings using information technologies. In this work it is shown how it is possible to join these concepts to create an infrastructure to facilitate the creation of applications that allows us the optimization of the energy resources. The aim of the project is improve comfort of the people who use a public building and, at the same time, improve the energy efficiency. To do that, it is designed a sensors and actuators network which allow the automation of systems that consume energy like light, heating, air conditioned, etc. So that, different spaces from a building are connected to the *Internet*, giving data about the current state from a specific area and allowing the creation of programs that automate processes that finally will be actuations in order to reduce energy consumption, without forgetting people comfort.

Índex de continguts

1. Introducció 6

1.1. Context ... 6

1.2. Actors implicats ... 7

1.3. Estat de l'art ... 8

1.3.1. *Internet* de les coses ... 8

1.3.2. Edificis intel·ligents ... 10

1.3.3. Solució que es proposa ... 11

2. Abast del projecte 12

2.1. Formulació del problema ... 12

2.2. Abast ... 13

2.2.1. Possibles obstacles ... 13

2.3. Metodologia i rigor ... 14

2.3.1. Anàlisi entorn ... 14

2.3.2. Requisits del maquinari ... 14

2.3.3. Simulació del sistema ... 14

2.3.4. Sistema físic ... 15

2.3.5. Anàlisi del sistema ... 15

3. Planificació temporal 16

3.1. Descripció de les tasques ... 16

3.1.1. Planificació del projecte i viabilitat ... 16

3.1.2. Anàlisis i disseny ... 16

3.1.3. Desenvolupament ... 17

3.1.4. Etapa final ... 17

3.1.5. Temps estimat ... 17

3.1.6. Recursos ... 18

3.2. Valoració d'alternatives i pla d'acció ... 18

4. Pressupost 20

4.1. Identificació de costos ... 20

4.1.1. Recursos materials ... 20

4.1.2. Costos directes per activitat ... 20

4.1.3. Costos indirectes ... 21

4.1.4. Amortitzacions ... 21

4.1.5. Contingències ... 22

4.1.6. Imprevistos ... 23

4.2. Estimació de costos 23
4.3. Control de gestió 24
5. Sostenibilitat 24
5.1. Puntuació 24
5.2. Dimensió econòmica 24
5.3. Dimensió social 25
5.4. Dimensió ambiental 25
6. Disseny del sistema 26
6.1. Esquema 26
6.2. Components 27
6.3. Funcionament pràctic 42
7. Conclusions 46
8. Treball futur 47
9. Revisió del projecte 48
9.1. Tecnologies existents utilitzades 48
9.2. Planificació final 48
9.3. Metodologia final 49
9.4. integració de coneixements 49
9.5. Lleis i regulacions 50
10. Referències 51
10.1. Bibliografia 51
11. Annex 53
11.1. Diagrama de *Gantt* inicial 53
11.2. Document control de gestió 54
11.3. Especificació *AS-XM1000* 55

Índex de taules

Taula 1. Dedicació estimada en hores per tasca 17
Taula 2. Utilització de recursos per activitat del Gantt 20
Taula 3. Costos i amortitzacions dels recursos materials 21
Taula 4. Costos dels recursos humans 22
Taula 5. Estimació del cost total del projecte amb amortitzacions 23
Taula 6. Estimació de la inversió total del projecte 24
Taula 7. Matriu de sostenibilitat de la planificació del treball 24

Índex de figures

Figura 1. Disseny d'edifici intel·ligent amb Internet de les coses 6
Figura 2. Internet de les coses, nascut entre els anys 2008 i 2009 9
Figura 3. Internet de les coses, la xarxa de les xarxes 10
Figura 4. Exemple d'edifici intel·ligent 11
Figura 5. Esquema del disseny 26
Figura 6. Node AS-XM1000 28
Figura 7. Codi d'inici del node 29
Figura 8. Codi d'execució automàtica segons el temps establert 29
Figura 9. Comanda execució sensor virtual 30
Figura 10. Fitxer de configuració sensor_conf_v4.txt 30
Figura 11. Codi sensor virtual 31
Figura 12. Comanda execució actuador virtual 32
Figura 13. Codi procés actuador virtual 32
Figura 14. Codi de recepció de les dades d'un actuador per part d'un sensor (virtuals) 33
Figura 15. Comanda execució interfície de connexió, demo.java 34
Figura 16. Codi funció principal demo.java 34
Figura 17. Codi funció de recepció de missatges del port sèrie, arxiu demo.java 35
Figura 18. Esquema plataforma servIoTicy 37
Figura 19. Contingut fitxer .json a enviar a la plataforma 37
Figura 20. Model format .json 38
Figura 21. Taula MySQL 39
Figura 22. Inserció nova fila a la base de dades 39
Figura 23. Comanda introducció camp position 40
Figura 24. Comanda introducció camp room 40
Figura 25. Comanda d'actualització del valor hum 40
Figura 26. Interfície de visualització web 41
Figura 27. Execució inicial interfície de connexió 42
Figura 28. Sensors reals connectats via USB i via antena ràdio 42
Figura 29. Execució sensor virtual 42
Figura 30. Recepció dades a la interfície de connexió 43
Figura 31. Execució actuador virtual 43
Figura 32. Canvi de rang de dades causat per un actuador 44
Figura 33. Taula base de dades actualitzada amb nou node 44
Figura 34. Visualització final 45

1. Introducció

1.1. Context

L'Internet de les coses, en anglès *Internet of things (IoT),* es refereix a una xarxa d'objectes de la vida quotidiana interconnectats. Qualsevol objecte que incorpori algun tipus d'element digital que doni informació podrà ser identificat per un equip informàtic de la mateixa manera que ho fa actualment un ésser humà.

Aplicar aquesta idea sobre un edifici consisteix en permetre que la gestió dels sistemes connectats a la xarxa elèctrica sigui controlada per un servidor en substitució d'una persona, com és habitual. Per exemple, una senzilla acció com és encendre el llum d'una sala, amb la infraestructura informàtica adient pot automatitzar-se en base la presència física de persones a l'espai en qüestió. L'automatització d'un edifici ens trasllada a un nou concepte, el dels edificis intel·ligents. [1][2]

Els edificis intel·ligents, en anglès *Smart buildings*, són aquells edificis equipats amb una infraestructura que permet l'automatització i control de la gestió del confort, la seguretat, la gestió energètica i el control d'accessos.

L'automatització d'un edifici consisteix en el control centralitzat automàtic dels sistemes de calefacció, aire condicionat, ventilació, llum, etc. Aquesta automatització té com a objectiu la millora del confort per part dels seus usuaris, l'eficiència operativa dels sistemes de l'edifici, la reducció del consum energètic i els seus costos.

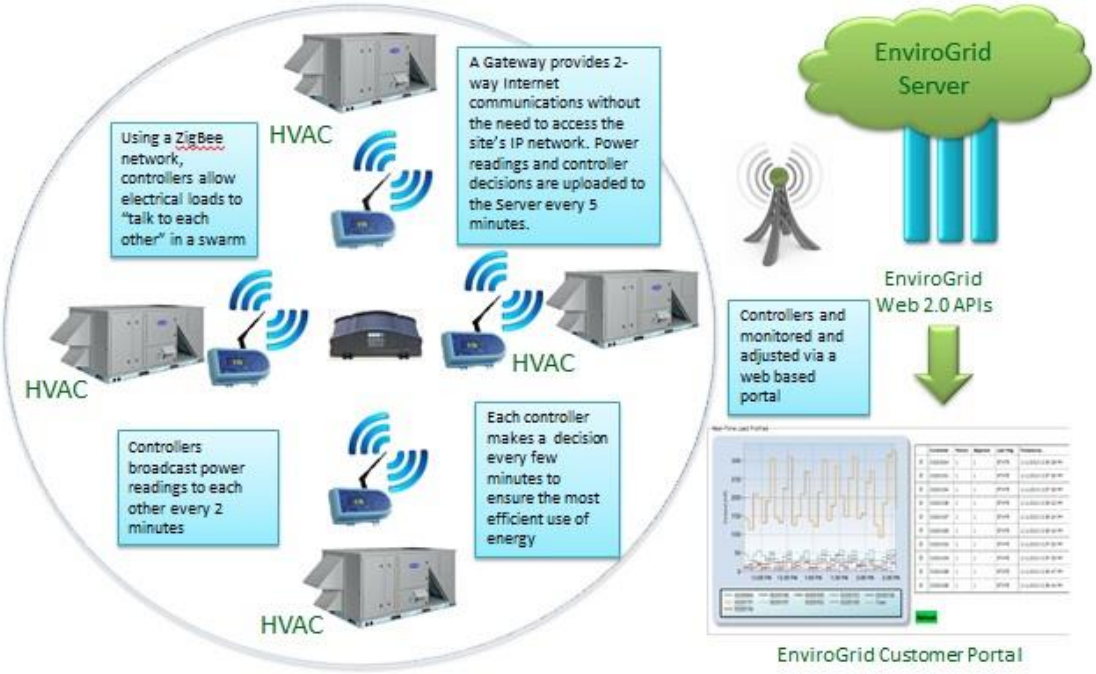


Figura 1 – Disseny d'edifici intel·ligent amb Internet de les coses

En la figura anterior es mostra un disseny de la infraestructura d'*Internet* de les coses aplicada en un edifici. Diferents sensors prenen dades de l'estat de diferents punts de l'edifici i transmeten la informació a un servidor que les tracta. De la mateixa manera s'utilitzen actuadors sobre els elements que modifiquen l'estat de l'edifici (llum, calefacció, etc). En aquest cas, es permet el control i monitorització de tot el sistema mitjançant un portal *web* que està connectat al servidor central. [4][5]

1.2. Actors implicats

La realització d'aquest projecte té diferents perfils implicats. El principal és el desenvolupador del projecte, però també hi haurà el rol de supervisor i tutor del projecte així com els usuaris finals del producte resultant.

Tots els rols estan relacionats i d'ells depèn la valoració de les necessitats que s'han de cobrir i quina és la millor manera d'arribar als objectius marcats.

- L'encarregat de desenvolupar el projecte serà l'autor d'aquesta planificació. Serà l'encarregat d'executar totes les fases del projecte, des de la planificació i l'estudi de l'edifici fins al disseny i implementació de la infraestructura necessària per al seu funcionament.

- El tutor del projecte serà qui supervisarà, com a enginyer, el desenvolupament de totes les fases del projecte i qui anirà guiant sobre el correcte rumb. Aconsellarà en la presa de decisions en les fases de disseny i controlarà el compliment del calendari i dels objectius marcats.

- La infraestructura que es creï un cop finalitzat el projecte formarà part d'un edifici públic on els seus usuaris se'n podran beneficiar. Per tant, l'objectiu d'aquest treball són totes aquelles persones que utilitzin l'edifici en concret on es portarà a terme el desenvolupament de la infraestructura.

Tant el desenvolupador com el tutor estaran al dia del projecte amb les reunions setmanals, on s'intercanviaran punts de vista per redirigir de la millor manera possible la seva execució.

Els usuaris podran gaudir del resultat final però se'ls haurà de tenir en compte durant les fases de proves per minimitzar-ne el seu impacte.

1.3. Estat de l'art

1.3.1. *Internet* de les coses

Segons el grup de solucions empresarials basades en *Internet* de l'empresa *Cisco, l'Internet* de les coses és senzillament el punt en el temps en el que es van connectar a *Internet* més "coses" o "objectes" que persones. [1][2]

En termes d'informàtica, es refereix a una xarxa d'objectes de la vida quotidiana interconnectats. El concepte és molt senzill però la seva aplicació és complicada. Si tots els objectes del nostre entorn disposessin d'un identificador que permetés la seva gestió a distància, aquests podrien ser gestionats per computadors de la mateixa manera que avui en dia ho són per humans.

L'origen de l'*Internet* de les coses es pot deure a l'Institut de Tecnologia de Massachusetts (*MIT*), concretament al treball de l'*Auto-ID Center*. Fundat l'any 1999, feia investigacions en el camp de la identificació per radiofreqüència en xarxa (RFID) i les tecnologies de sensors emergents. Els laboratoris d'investigació eren formats per set universitats de quatre continents diferents.

L'any 2003 hi havia uns 6,3 mil milions de persones al planeta, i hi havia 500 milions de dispositius connectats a *Internet*. Dividint la quantitat de dispositius connectats per la població mundial, obtenim un resultat que ens indica que hi havia menys d'un dispositiu per persona (índex del 0,08). Segons la definició de *Cisco,* l'any 2003 encara no existia l'*Internet* de les coses.

L'explosiu creixement de s*martphones* i tauletes va elevar a 12,5 mil milions l'any 2010 la quantitat de dispositius connectats a *Internet*, mentre que la població mundial va ascendir als 6,8 mil milions de persones, pel que per primera vegada en la història el nombre de dispositius connectats per persona és superior a 1 (1,84 exactament).

Segons el grup d'investigació de *Cisco,* el naixement de l'*Internet* de els coses es troba en algun punt entre els anys 2008 i 2009.

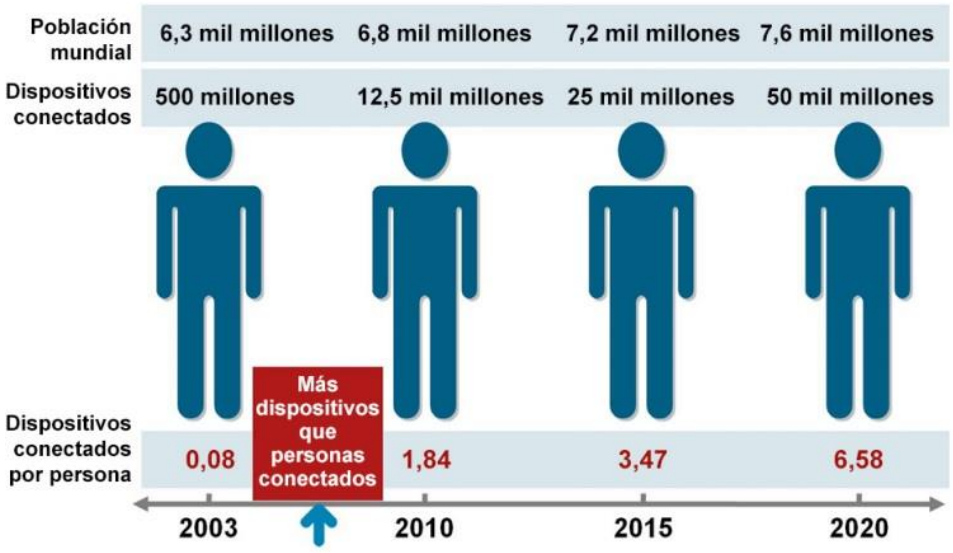


Figura 2 – Internet de les coses, nascut entre els anys 2008 i 2009

De cara al futur, el grup de recerca de *Cisco* preveu que hi haurà 25 mil milions de dispositius connectats l'any 2015 i 50 mil milions al 2020. Aquestes xifres es basen en les dades actualment vàlides, no es té en compte l'evolució tecnològica.

També s'ha de tenir en compte que el càlcul es basa en el total de la població mundial, no només en la part que realment té accés a *Internet*. Si només es tingués en compte les persones que realment hi tenen accés, les xifres creixerien dràsticament. [1]

Actualment l'*Internet* de les coses està compost per un conjunt dispers de diferents xarxes amb diferents finalitats. No és el mateix la xarxa de què disposa un automòbil per controlar tots els seus paràmetres que la que pot tenir un edifici comercial o residencial.

A mesura que l'*Internet* de les coses evolucioni, aquestes xarxes i moltes altres estaran connectades amb la incorporació de capacitats de seguretat, anàlisis i administració. Per tant, totes aquestes xarxes dispars seran unides en un futur. [1]

El punt en què ens trobem avui en dia és l'inici de la unió de diferents sistemes ja existents per aconseguir la comunicació i intercanvi d'informació entre tots ells. Això permetrà augmentar les funcionalitats que cada un d'aquests sistemes pot arribar a tenir de manera separada.

L'Internet de les coses és la unió de tot allò que pugui ser connectat a la xarxa i controlat mitjançant *Internet* des de qualsevol lloc.

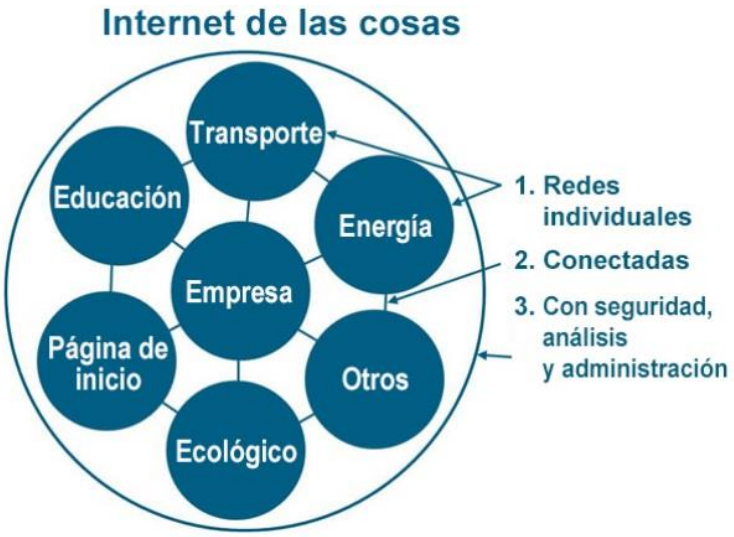


Figura 3 – Internet de les coses, la xarxa de les xarxes

1.3.2. Edificis intel·ligents

Un edifici intel·ligent es defineix com una estructura que ofereix als seus usuaris i administradors un conjunt coherent d'eines i facilitats. Està dissenyat per cobrir tots els possibles avenços tecnològics, tenint en compte les necessitats reals dels usuaris i administradors de l'edifici.

La finalitat d'un edifici intel·ligent és la de proporcionar un ambient de confort i seguretat, per maximitzar la productivitat i creativitat així com fer que la gent es senti a gust en el seu interior. A més, aquest tipus d'edificis han de proporcionar mitjans de manteniment eficients, minimitzant els costos. [5]

Històricament la humanitat ha construït edificis per crear un entorn controlat on viure i treballar. Al llarg de les dècades però, ha canviat les prioritats en el disseny i la organització dels edificis, especialment en el cas de les oficines.

Actualment, la concepció d'un edifici des de la seva etapa de planificació té en compte tots els elements que posteriorment serviran per disposar d'un ambient més productiu, minimitzant els costos.

Sota aquest concepte sorgeixen els edificis intel·ligents. Aquest nom va ser el resultat de la integració de diverses tecnologies, bàsicament els computadors i sistemes de comunicació.

Els edificis actuals s'han vist sotmesos a intensos estudis orientats a crear ambients ergonòmics que ofereixin una gran quantitat de serveis i facilitats. A més, aquests estudis tenen un argument comercial, ja que els fan més atractius de cara a possibles compradors.

Els edificis intel·ligents van aparèixer a mitjans de 1980. Inicialment el terme "intel·ligent" feia referència a l'alt grau d'automatització i la integració del tots els sistemes. A dia d'avui aquest terme és molt popular i s'ha estès per tot el món. [5]

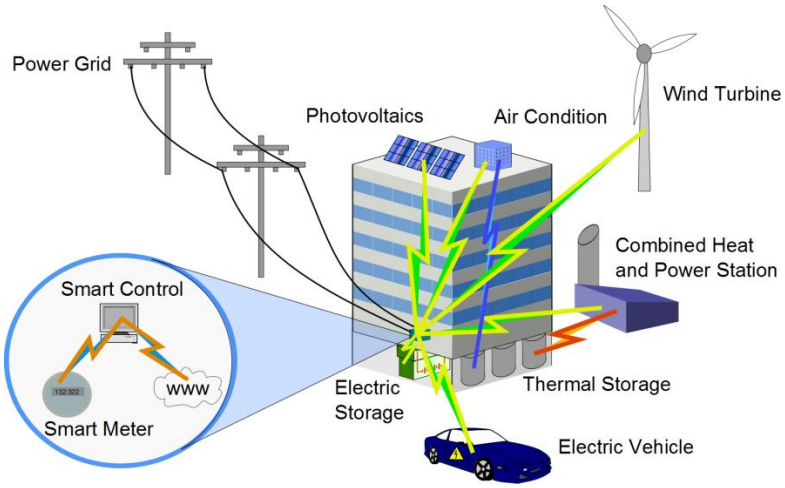


Figura 4 – Exemple d'edifici intel·ligent

1.3.3. Solució que es proposa

La majoria dels edificis actuals no disposen de cap sistema que permeti automatitzar el control dels sistemes que generen consum energètic. Els dissenys antics no tenien en compte la possibilitat d'incorporar la tecnologia com a mètode per a l'eficiència energètica.

L'enllumenat, la calefacció o l'aire condicionat són alguns dels sistemes que requereixen la mà de l'ésser humà per activar-se i desactivar-se, sent aquest un risc pel que fa a l'eficiència energètica de l'edifici. Si tots aquests sistemes fossin controlats per un sistema informàtic que s'adaptés a les necessitats de l'usuari, aleshores tots ells tindrien un ús optimitzat i no patirien una despesa innecessària.

Per tant, el principal problema que tenim en l'actualitat és la falta d'un control optimitzat del consum energètic dels diferents sistemes que permeten un major confort a l'usuari d'un edifici.

Mitjançant el concepte de l'*Internet* de les coses, en aquest projecte es desenvoluparà una infraestructura que permeti optimitzar l'ús dels sistemes que generen consum energètic en un edifici.

Aquesta infraestructura estarà formada per diferents sensors que seran capaços de captar informació del seu entorn i actuar en conseqüència.

Es disposarà d'un servidor central que captarà la informació generada a partir dels sensors repartits per diferents zones de l'edifici i en base a aquesta generarà informació de resposta. La resposta serà rebuda per actuadors que s'encarregaran de realitzar les accions que permetran reduir el consum energètic.

Tindrem doncs, un sistema capaç de recollir diferents dades del seu entorn i actuar-ne en conseqüència de forma autònoma, sense la necessitat d'interacció directe amb cap persona.

Ja que cada edifici té les seves característiques funcionals i arquitectòniques particulars, es considera que la millor opció és dissenyar un nou sistema a mida per a l'àmbit d'actuació.

2. Abast del projecte

2.1. Formulació del problema

El desenvolupament d'una infraestructura *Internet of things* (IoT) forma part d'un projecte més gran que vol crear un innovador ecosistema informàtic per donar suport als ciutadans per tal d'aconseguir una gran eficiència energètica en els edificis públics.

El problema plantejat consisteix en analitzar un entorn físic tancat i actuar-hi de tal manera que es redueixi el seu consum energètic. L'objectiu és adaptar els sistemes que generen aquest consum (llum, calefacció,...) segons les necessitats dels usuaris de l'edifici per donar-hi confort però a la vegada reduir-lo i fer-lo més sostenible. Per exemple, es necessitarà un sistema capaç d'encendre o apagar els llums d'una zona on hi pugui haver algú.

La creació d'un sistema capaç d'interactuar amb el seu entorn requereix d'una xarxa de sensors i actuadors estesos per l'àmbit d'actuació i d'un punt de recollida i tractament de les dades que s'adquireixen, un servidor.

Degut a l'ànima del projecte (la sostenibilitat energètica), els elements a utilitzar hauran de disposar d'un maquinari de baix consum així com un programari optimitzat al seu ús. Per tant, podem dividir el projecte en dos apartats:

- Selecció de components físics de baix consum. La xarxa de sensors que s'estendrà en l'edifici de proves funcionarà amb bateries, havent d'optimitzar al màxim el seu ús.
- Disseny i programació de la xarxa de comunicació entre sensors, servidor i actuadors. Cal determinar l'estructura de la xarxa a muntar així com el sistema de comunicació i captació de dades que s'enviaran al servidor.

El projecte consisteix doncs en analitzar, avaluar i seleccionar les plataformes *hardware* més adients per a la xarxa de sensors que requereixen, principalment, un molt baix consum i una interacció directa amb l'entorn. Aquestes plataformes hauran de ser capaces de captar i transmetre dades de temperatura, humitat i lluminositat fent ús d'algun protocol sense fils de baix consum.

A més, caldrà dissenyar la xarxa de comunicació entre els diferents components del sistema. Els principals elements d'aquesta xarxa seran els sensors encarregats de captar dades de l'entorn, els actuadors que seran capaços de modificar els paràmetres ambientals de l'edifici i el servidor que els unirà tots dos. Caldrà estudiar quina arquitectura requereix la xarxa i quins protocols de comunicació seran els més adients.

2.2. Abast

Per poder escollir correctament el *hardware* adient per al sistema primer haurem d'analitzar quin serà l'àmbit d'actuació. Totes les proves d'aquest projecte es realitzaran a l'edifici D6 del Campus Nord de la Universitat Politècnica de Catalunya a Barcelona. Aquest edifici públic està format per diferents zones com aules per a classes, despatxos individuals de professors, aules de treball per a grups o sales de reunions.

Inicialment caldrà estudiar cada un dels espais per determinar quin tipus de sensor és necessari. Una zona més amplia requerirà d'un sensor més potent i haurà de captar més dades, ja que hi haurà més factors que podran influir en els canvis en l'ambient. Més espai segurament implicarà més flux de persones i per tant més canvis en l'entorn. Si pel contrari l'espai és petit, les variacions possibles seran més petites i requeriran menys esforç energètic.

Un cop determinats els espais on es realitzaran les proves i els tipus de sensors que s'utilitzaran, es procedirà al seu desenvolupament. Per fer-ho serà necessari disposar d'un maquinari preparat, pel que caldrà esperar a la seva adquisició i recepció.

Mentre no es disposi del maquinari físic, es realitzarà una primera fase on els components seran emulats. Mitjançant processos virtuals en un ordinador, es simularà el comportament dels sensors i actuadors per poder disposar d'un sistema on fer les primeres proves de disseny de la xarxa i tipus de comunicació.

Serà necessari disposar de tres tipus de processos, un que simuli el comportament de cada component real (sensor, actuador i servidor). Cada un d'ells haurà d'enviar i rebre dades segons les funcions reals que representen.

La primera prova del sistema real es realitzarà una vegada es disposi dels sensors, que hauran de ser configurats seguint el disseny representat amb les proves simulades. En una primera prova, s'utilitzarà un únic sensor que segons les dades rebudes se li hauran d'anar ajustant paràmetres com el temps de captació de les dades per obtenir la millor relació entre la informació obtinguda i la vida de les bateries.

Es posaran en funcionament altres unitats per tal de comparar resultats i extreure la millor configuració possible. A més, hi haurà la possibilitat de que cada una de les unitats es comuniqui amb algunes de les més properes per oferir informació i ser configurades o per fer de pont en l'enviament de dades al servidor. Aquesta última característica s'anirà estudiant la seva necessitat al llarg del desenvolupament.

2.2.1. Possibles obstacles

El principal element en el sistema són les unitats que disposen dels sensors. Degut a que aquestes es troben disperses per diferents llocs i que s'ha decidit fer-les funcionar amb bateries, ens podem trobar amb el problema d'un ràpid esgotament de la seva vida. Aquest serà el punt més crític i important del projecte. Haver de substituir les bateries amb massa freqüència va en contra de l'objectiu principal, la sostenibilitat energètica.

En un inici la prova física del sistema es realitzarà amb poques unitats amb sensors, pel que ens podem trobar amb una falta de dades captades que no donin prou informació al servidor. Si no es capten prou dades, no es pot actuar òptimament sobre els actuadors que seran els que realment modifiquin el ecosistema de l'edifici. No podem apagar el llum de totes les aules si no disposem de suficients sensors per detectar la presència de persones a totes elles.

Un altre problema que podem tenir és l'excés de dades captades i el seu procés d'emmagatzematge. Els sensors capten dades amb una freqüència establerta i s'ha de decidir que fer amb elles, si conservar-les o descartar-les. No podem guardar la totalitat de dades generades ja que requeririen un espai excessivament gran i que realment no seria necessari però a la vegada s'han de tenir les suficients com per evitar el problema d'escassetat.

Per tant, es corre el risc de no reduir com caldria el consum energètic de l'edifici a causa d'una mala gestió dels recursos del sistema que es vol implementar. En el pitjor dels casos, podria empitjorar. Per això caldrà tenir cura en la selecció dels components *hardware* i del disseny *software* i de la xarxa.

2.3. Metodologia i rigor

2.3.1. Anàlisi entorn

Per a la creació de la infraestructura formada pels sensors, servidor i actuadors, primer de tot haurem de disposar d'un banc de proves accessible. Com que l'edifici forma part de la universitat la facilitat d'accés està assegurada.

El següent pas serà analitzar els espais de que es disposen. Serà necessari determinar quines zones permetran obtenir més dades rellevants per al sistema. Ens pot interessar una aula amb major afluència de públic i horaris coneguts però també un despatx individual on la presència de persones no és sempre previsible. Això donarà riquesa i varietat d'informació.

2.3.2. Requisits del maquinari

Conegut l'entorn, el següent pas serà l'adquisició del maquinari a utilitzar. El necessari seran els sensors. Per fer-ho s'haurà de determinar primer de tot quins són els factors ambientals de l'edifici a tenir en compte. Inicialment volem mesurar la humitat, temperatura i intensitat de la llum en cada un dels punts on s'instal·lin els sensors. Necessitarem doncs, un aparell amb aquests tres tipus de captadors de dades de l'ambient.

2.3.3. Simulació del sistema

Durant l'espera d'arribada del material, es començarà la implementació de la infraestructura virtualment. En aquest punt es crearan, en un sol ordinador, diferents processos que representin els components físics reals.

El primer representarà un sensor que serà capaç d'informar sobre la temperatura, la humitat i la intensitat de la llum. A continuació es connectarà amb un segon procés que simularà el servidor i rebrà les lectures. Finalment el tercer procés representarà un actuador que s'activarà en funció de les dades rebudes del servidor.

Aquest sistema virtual serà un sistema de proves de cara al disseny final de la infraestructura. Es podria considerar doncs, una eina per al disseny de la xarxa.

2.3.4. Sistema físic

Ja disposant d'un maquinari bàsic i d'un sistema virtual de referència, la següent tasca consistirà en la substitució dels processos emulats pel maquinari real.

S'haurà d'adaptar el sistema virtual als requisits i característiques dels nous components. El tipus de connexions que s'utilitzen entre processos virtuals són diferents dels reals i segons el model de sensor escollit les lectures poden diferir en la forma d'obtenir-les.

La comunicació de la xarxa de sensors serà amb un sistema inalàmbric de baix consum tipus *bluetooth*. Aquest sistema és de curt abast, pel que s'haurà d'estudiar la conveniència d'utilitzar la xarxa de sensors com a pont entre ells mateixos per arribar al servidor o d'altres punts de convergència que facin de passarel·la cap al servidor. La gestió d'aquesta xarxa també és molt important, ja que si no es gestiona bé es poden perdre dades.

2.3.5. Anàlisi del sistema

Finalment, caldrà comprovar com funciona tota la infraestructura. Amb totes les unitats instal·lades als seus llocs corresponents, es verificarà des del servidor que totes elles envien dades i que ho segueixen fent durant el temps que s'hagi calculat de duració de les bateries. Si és necessari, s'utilitzaran instruments alternatius de mesura de la temperatura i humitat (com un termòmetre comercial comú) per garantir la correcta lectura.

Les dades preses seran estudiades, comparades entre elles i amb les mesurades amb aparells alternatius, per veure que en diferents condicions el sistema segueix funcionant. Hem d'obtenir lectures correctes en dies laborables, festius, hores de tancament de l'edifici, nocturnes, amb diferent quantitat de persones a cada zona, etc.

Si es detecta una falta de punts de presa de dades, per poder provar com treballen tots els components quan hi ha gran quantitat de sensors enviant dades, aleshores s'aprofitaran els processos virtuals comentats anteriorment per generar un major flux de dades en el sistema.

3. Planificació temporal

3.1. Descripció de les tasques

3.1.1. Planificació del projecte i viabilitat

La primera tasca que s'ha de realitzar en el projecte consisteix en la seva planificació i l'estudi de la viabilitat. Primer de tot cal determinar quin és l'abast a partir del problema formulat inicialment. A continuació cal extreure totes les tasques que són necessàries per la realització del projecte, així com establir una metodologia de treball que garanteixi el compliment dels objectius plantejats.

Determinat el pla a seguir i les tasques a realitzar, el següent pas consisteix en establir el pressupost que necessita el projecte segons les hores i recursos necessaris així com els seus costos associats.

L'últim pas en la planificació és conèixer l'estat de l'art i disposar amb ell d'un coneixement que permeti tenir un punt de partida més sòlid i ajudi a decidir la viabilitat del projecte. Per això s'ha de conèixer quines solucions similars hi ha al mercat i veure que ens poden aportar.

3.1.2. Anàlisis i disseny

Aquesta etapa requereix un estudi dels diferents àmbits que es tractaran durant el projecte així com el disseny del pla d'execució.

Principalment s'haurà de veure quins són els requisits que s'han de complir. Aquest conjunt de requisits permetran assolir els objectius marcats, pel que seran la base de l'èxit del projecte. Cada un dels objectius haurà de poder complir-se amb els requeriments que s'estableixin per a ell en concret.

Establerts els requisits necessaris, el següent pas consistirà en dissenyar un pla d'execució de les tasques a desenvolupar a la fase d'implementació. Aquest disseny haurà d'incloure tant els sistemes *hardware* com el *software.* Per tant, es començarà per l'anàlisi de tot el maquinari que s'utilitzarà i es continuarà amb el programari de que disposarà el sistema. Cal destacar que l'ús de *hardware* específic i de baix consum crea per ell mateix unes limitacions de recursos que el disseny del *software* i de la xarxa que s'ha de tenir sempre en compte.

Cal afegir, a més, que un cop obtingudes les funcionalitats requerides pel sistema, s'haurà d'establir un pla de verificació de totes elles així com un sistema de proves per determinar els ajustaments necessaris. Aquestes proves s'aniran establint segons l'evolució de tota la infraestructura.

Per tant, el disseny del projecte es basa en tres punts: *hardware, software* i sistemes de verificació.

3.1.3. Desenvolupament

La fase de desenvolupament del projecte la podem dividir en els següents apartats:

- Adquisició sensors. El primer pas consisteix en l'anàlisi de diferents components *hardware* existents al mercat i la selecció del més adient. S'haurà de dedicar un temps específic per escollir el que més s'adapti als requisits del projecte.
- Estudi de l'entorn de treball i configuració. Disseny i implementació del sistema emulat.
- Estudi i disseny del sistema físic. En base al material adquirit, cal estudiar el seu funcionament a nivell tècnic. Cada fabricant pot precisar dels seus propis programes informàtics específics per al seu funcionament (compilador, instal·lador, etc). Per tant, serà necessari aprendre a utilitzar-los i configurar-los. A part d'aquests programes, també serà indispensable conèixer en profunditat les característiques del *hardware*, pel que s'hauran d'estudiar els manuals dels sensors.
- Implementació *hardware* i *software*. Aquest apartat consisteix en el disseny i muntatge de la infraestructura, des de les connexions entre sensors i servidor fins disseny i programació de tot el codi a implementar.
- Verificació sistema i ajustaments. L'última fase del desenvolupament consistirà en la creació de les proves per a la validació del funcionament de tota la infraestructura i, segons els seu resultats, en la seva configuració.

3.1.4. Etapa final

Amb tot el sistema implementat i disposant de la infraestructura real en funcionament, quedarà per fer l'anàlisi dels resultats i la corresponent extracció de conclusions. Els resultats hauran de satisfer els requisits inicials que mostraran que els objectius s'han assolit exitosament. El treball realitzat serà documentat i finalment presentat.

3.1.5. Temps estimat

La durada màxima del projecte és de tres mesos i mig. S'inicia el dia 16 de Febrer de 2015 i ha d'estar acabat i entregat a dia 29 de Maig de 2015. La dedicació estimada inicialment per cada tasca és la següent:

Tasca	Dedicació en hores
Planificació del projecte i viabilitat	40
Anàlisis i disseny	60
Desenvolupament	200
Etapa final	50
Total	**350**

Taula 1 – Dedicació estimada en hores per tasca

En l'estimació anterior es considera una dedicació de 20 hores setmanals amb puntes de 30 hores setmanals per la meitat de la fase de desenvolupament (veure diagrama de *Gantt* en Annex).

3.1.6. Recursos

Dividirem els recursos del projecte en tres tipus:

- Humans. El projecte serà desenvolupat per l'estudiant col·laborador amb el grup de recerca *Communications and Broadband Architecture* (CBA), del departament d'arquitectura de computadors de la Facultat d'Informàtica de Barcelona. L'estudiant rebrà suport i serà supervisat pels director i codirector del projecte, Davide Careglio i Sergio Ricciardi respectivament, membres el CBA.
- *Hardware.* Els recursos *hardware* seran formats per almenys un ordinador que permeti la programació de tot el *software* necessari, un servidor del departament d'arquitectura de computadors de la facultat i tot el maquinari de sensors necessaris. Aquest maquinari està per especificar i encara no se'n coneixen les característiques concretes. Tot depèn de la fase d'anàlisi i posterior selecció dels sensors.
- *Software.* Pels mateixos motius esmentats anteriorment, fins a l'adquisició del material no es podrà determinar quin programari específic es necessita. El que ja s'està utilitzant són les eines de gestió i documentació del projecte. El diagrama de *Gantt* utilitza l'eina *Gantter,* la redacció del document es fa amb *Microsoft Office 2010* sobre el sistema operatiu *Windows 7.* A més, s'utilitza *Dropbox* per facilitar l'accés dels documents i arxius des de diferents punts així com a còpia de seguretat d'aquests.

A mesura que es vagin coneixent els recursos a utilitzar, aquests seran afegits i detallats en la documentació del projecte. Un altre recurs a destacar que no forma part dels tipus anteriors és l'edifici D6 del Campus Nord, que serà el camp de proves de tota la infraestructura.

3.2. Valoració d'alternatives i pla d'acció

Entre totes les etapes del projecte la que pot generar desviacions respecte el pla inicial és la fase de desenvolupament. El primer problema el podem tenir només començar, en el moment d'adquirir el material. Els sensors necessaris es compraran per *Internet* i els terminis de lliurament poden ser més llargs del previst. Ja que sense el material no es pot iniciar el projecte real, hi ha una dependència total de la recepció dels components.

En un principi es preveu un marge d'entrega de fins a 30 dies des del dia de l'encàrrec. Per a cada dia extra, haurem d'augmentar el temps de treball previst per les fases posteriors 4 hores. És a dir, cada jornada perduda implicarà afegir les hores equivalents com a extres del projecte i d'aquesta manera s'evitarà retardar el dia de lliurament final.

Com a mesures preventives l'únic que podem fer és avançar el dia de la demanda de material per així disposar posteriorment de més temps en cas de retard. Durant el temps d'espera de l'entrega es treballarà sobre el sistema emulat, aprofitant així el temps total del projecte.

En la definició dels possibles obstacles es destacava la necessitat d'un bon ús de les bateries que els sensors tindran. En aquest aspecte, si es detecta un ràpid esgotament, s'haurà de replantejar el disseny del *software* per reduir el consum energètic. Això implicarà realitzar menys accions, pel que segurament la precisió o la freqüència de lectura de dades serà disminuïda.

La configuració de nou del sistema es considera inclosa dins la fase d'implementació del sistema real ja que és altament probable que es necessiti. No es considera doncs la necessitat d'invertir més recursos dels previstos inicialment.

La prova a l'edifici es farà amb poques unitats de sensors. Un inconvenient que això pot suposar és la falta de captació de suficient varietat de dades, ja que hi haurà poca varietat de mostres. Tot i que es podrà comprovar el funcionament de la infraestructura, podria ser que no es disposés de dades suficients com per obtenir un anàlisis que doni informació suficient.

En aquest cas no es considera la possibilitat d'equipar més unitats pel sobre cost excessiu que podria representar, així que l'única alternativa seria la de desplaçar manualment els sensors i anar obtenint les dades en zones més petites però que proporcionalment representés una major quantitat de sensors. L'estudi aleshores seria subdividit en zones més petites. Les zones que ja se'n coneixen dades i no disposen de sensor passarien a ser emulades.

Per tant, no es considera la possibilitat d'invertir en més material, sinó treballar amb el que es disposa per captar noves dades i utilitzar emuladors per generar dades en base a les captades en una fase anterior per un sensor real.

Com que l'objectiu és el funcionament del sistema, amb els sensors adquirits inicialment i els emuladors es preveu poder realitzar el projecte. Si no és així i realment es necessiten més sensors físics, cal tenir en compte que l'adquisició de cada un d'ells té un cost aproximat de 90€ (com es veurà en l'apartat de costos) i el seu temps d'entrega estimat és de 30 dies.

Un altre problema que podem trobar és l'excés de dades llegides. En la fase de desenvolupament es té en compte que no es poden guardar totes les dades llegides per sempre, així que s'hi té en compte una política d'esborrat de les dades més antigues. D'aquesta manera s'evita una major necessitat d'espai d'emmagatzematge.

Si no tenim espai on guardar les dades més recents, aleshores aquestes es perdran i el sistema podria deixar de funcionar. En aquesta cas s'hauria de dissenyar un protocol d'emmagatzematge especial, que implicaria una nova tasca a realitzar en paral·lel a la fase de desenvolupament amb la inversió extra d'hores segons la seva complexitat.

Qualsevol altre imprevist que demandi major temps de desenvolupament s'haurà de resoldre en les puntes de dedicació setmanal (fins a 30h) que s'han previst en l'estimació anterior. En cas contrari podríem patir un retard en la finalització del projecte.

Ja que el temps de desenvolupament el portarà a terme l'estudiant, totes les hores extres no suposaran un cost econòmic extra.

4. Pressupost

4.1. Identificació de costos

4.1.1. Recursos materials

A. *HP Pavilion dv6 Notebook PC*
B. Servidor DAC (departament arquitectura de computadors)
C. Sensors (pendent d'especificar tipus i model)
D. *Microsoft Windows 7 Professional*
E. *Ubuntu 14.04 LTS*
F. *Microsoft Office 2010*
G. *Gantter*
H. *Dropbox*
I. Editor de text *Gedit*
J. Compilador *GCC*

4.1.2. Costos directes per activitat

Activitat	Recursos
1. Planificació del projecte i viabilitat	
1.1. Formulació i abast del problema	A, D, F, H
1.2. Planificació temporal	A, D, F, G, H
1.3. Pressupost del projecte	A, D, F, H
1.4. Presentació preliminar	A, D, F, H
1.5. Context i bibliografia	A, D, F, H
1.6. Documentació final i presentació	A, D, F, G, H
2. Anàlisi i disseny	
2.1. Definició dels requisits i objectius	A, B, C, E, H, I, J
2.2. Disseny hardware	A, B, C, E, H
2.3. Disseny software	A, B, C, E, H, I, J
2.4. Disseny sistema de verificació	A, B, C, E, H, I, J
3. Desenvolupament	
3.1. Adquisició de sensors	A, D
3.2. Entorn de treball i configuració	A, B, C, E, H, I, J
3.3. Implementació	A, B, C, E, H, I, J
3.4. Verificació sistema	A, B, C, E, H, I, J
4. Etapa final	
4.1. Documentació	A, D, F, G, H
4.2. Presentació	A, D, F, H

Taula 2 – Utilització de recursos per activitat del Gantt

4.1.3. Costos indirectes

Les despeses generals i impostos no es comptabilitzen en el cost del projecte ja que tot es desenvolupa dins de la infraestructura de la universitat. Això implica que no es pugui extreure una part de la despesa de l'edifici on es desenvolupa el treball que sigui atribuïble al projecte. Pel mateix motiu tampoc s'hi poden relacionar impostos, ja que tant els recursos humans com els materials s'engloben dins la universitat.

4.1.4. Amortitzacions

A continuació es detallen els costos associats als recursos materials. Cada recurs inclou la corresponent amortització en funció del temps de vida previst. L'amortització es calcula tenint en compte que es realitzarà en 4 anys amb una jornada laboral de 4h diàries. El càlcul serà: Preu x hores projecte / (4 anys x 232 feiners/any x 4 h/dia)

Recurs	Preu	Temps de vida	Amortització
HP Pavilion dv6 Notebook PC	850€	4 anys	80.15€
Servidor DAC	--	--	--
Sensors	90€ x 3 *	4 anys	25.46€
Microsoft Windows 7 Professional	0€ **	4 anys	--
Ubuntu 14.04 LTS	0€	--	--
Microsoft Office 2010	79€ **	4 anys	7.45€
Gantter	0€	--	--
Dropbox	0€	--	--
Editor de text *Gedit*	0€	--	--
Compilador GCC	0€	--	--
Total	1199€		113.06€

Taula 3 – Costos i amortitzacions dels recursos materials

*El preu aproximat es calcula de 90€ per sensor. Inicialment es preveu fer les proves amb 3 sensors físics. El preu final depèn de la elecció del model de sensor, encara per determinar.

**El sistema operatiu *Windows 7* està disponible gratuïtament per a tot estudiant de la Facultat d'Informàtica de Barcelona. El paquet d'ofimàtica *Microsoft Office* disposa de preu especial reduït per estudiants universitaris.

El servidor del DAC és una infraestructura pública, que forma part de la UPC, pel que no es comptabilitza el seu preu per utilitzar-lo.

A la següent taula es detalla la dedicació en hores i el seu cost dels diferents rols que participen en el projecte.

Rol	Hores	Preu/hora	Cost
Enginyer cap de projecte	20 h	50€/h	1000€
Estudiant col·laborador	330 h	0€/h	0€
Total	350 h		1000€

Taula 4 – Costos dels recursos humans

El projecte serà desenvolupat per l'estudiant i el cap de projecte s'encarregarà de supervisar la feina amb reunions periòdiques setmanals.

4.1.5. Contingències

En aquest apartat s'exposen els possibles obstacles que poden endarrerir el projecte o forçar-ne modificacions i com afecta als recursos destinats.

- Excessiu consum de les bateries dels sensors. Per a un ús d'aquest tipus, s'estima una duració de fins a 2 anys de les piles. Per aquest problema la solució passa per redissenyar el sistema *software* que en gestiona els recursos. Tornar a dissenyar el *software* suposaria una dedicació extra per tal d'analitzar, detectar i corregir els errors de disseny.
- Escassetat de dades. Pot ser per la falta de freqüència en la captació de dades per part dels sensors, el que requeriria modificar la configuració del sistema, o bé per la falta de punts de presa de dades. Cada nova unitat té un preu aproximat final de 90€ i un temps d'entrega per part del fabricant (entre 20 i 30 dies).
- Excés de dades a emmagatzemar. Tampoc volem una freqüència massa elevada de presa de dades ja que aquestes han de ser emmagatzemades i l'espai per fer-ho és finit. Aquí s'aplica un protocol d'esborrat de dades, on inicialment el que es fa és esborrar les més antigues (es guarden les dues últimes setmanes). Si aquest sistema no és suficient, ja sigui per capacitat de memòria o per necessitat de més dades, aleshores s'haurà d'estudiar un nou protocol.
- No reducció del consum. Per qualsevol dels motius anteriors ens podem trobar que generem més cost amb el nostre sistema que el que aquest estalvia en consum energètic. Si això passa, haurem de dedicar un temps a l'anàlisi del problema per saber quin és el factor a corregir.

4.1.6. Imprevistos

- Caiguda servidor. Aquest problema no es pot solucionar directament, cal esperar a que el servei de manteniment del servidor el repari. Caigudes d'aquest tipus poden suposar una aturada del sistema de dies, no més de tres o quatre habitualment.
- Fallada de l'ordinador de desenvolupament. El projecte es desenvolupa en un sol ordinador i si aquest falla s'ha de reparar o bé substituir-lo per un altre. Si la reparació la pot realitzar el propi desenvolupador del projecte, s'estima un temps de 3 o 4 dies entre adquirir la possible peça a substituir (preu inferior a 100€, segons la peça) i configurar de nou la màquina. Si es decideix canviar de màquina, aquesta serà subministrada en préstec fins la finalització del projecte pel grup de recerca CBA. En aquest cas la configuració del sistema portarà uns 3 dies fins tornar a l'estat previ.
- Fallada sensors. Si no respon i es creu que és imprescindible per la continuació del projecte, aleshores se'n haurà d'adquirir un de nou que el substitueixi. La despesa serà la dels 90€ (aproximat segons model final) d'un nou sensor i els 20 o 30 dies de temps d'entrega més la hora de configuració.
- Fallada connexions xarxa o desconfiguració de la infraestructura. Requerirà un temps d'anàlisi per determinar el problema i unes hores de dedicació per solucionar-lo. Si és degut a una fallada d'un sensor, ens trobem en el cas esmentat anteriorment. Si es deu a la necessitat de configurar de nou el sistema, les 8 hores corresponents a dues jornades haurien de ser suficients per solucionar-ho o per descobrir un altre problema.
- Pèrdua dels avenços durant el desenvolupament. No suposaria un problema ja que qualsevol avanç serà guardat en disc extern com a còpia de seguretat i/o al núvol mitjançant *Dropbox*.

4.2. Estimació de costos

A continuació es mostra el cost total del projecte. El cost dels recursos materials va en funció de les seves hores d'utilització (les 350 previstes del projecte).

Tant en els imprevistos com en les contingències es comptabilitza la substitució/adquisició d'un nou sensor, amb un cost de 90€. També s'assumeix un cost extra de 100€ per a la possible substitució d'una peça a l'ordinador de desenvolupament.

Concepte	Cost
Recursos materials	113.06€
Recursos humans	1000€
Contingències (5%) – directes i indirectes	53.60€
Imprevistos	15.80€
Total	1183.06€

Taula 5 – Estimació del cost total del projecte amb amortitzacions

En la següent taula es mostra la inversió econòmica necessària per la realització del projecte. No es tenen en compte les amortitzacions ja que el material a comprar genera la despesa de cop.

Concepte	Cost
Recursos materials	1199€
Recursos humans	1000€
Contingències	190€
Imprevistos	90€
Total	2479€

Taula 6 – Estimació de la inversió total del projecte

4.3. Control de gestió

Cada setmana es realitza una reunió de seguiment i supervisió amb el cap del projecte. Amb ella s'analitza l'estat del projecte per preveure necessitats futures i anticipar-se, ja sigui per la necessitat d'adquirir nou material, detectar errors de disseny o fer modificacions en el plantejament del problema per millorar la solució del moment.

A més, es disposarà d'un document per anar fent el seguiment del material utilitzat i de les tasques finalitzades, en procés i pendents. Aquest document serà actualitzat segons les modificacions del projecte (veure document de control de gestió en Annex).

5. Sostenibilitat

5.1. Puntuació

Sostenible?	Econòmica	Social	Ambiental
Planificació	Viabilitat econòmica	Millora qualitat de vida	Anàlisi de recursos
Valoració	7	8	8

Taula 7 – Matriu de sostenibilitat de la planificació del treball

5.2. Dimensió econòmica

Des del punt de vista econòmic, un dels objectius del projecte consisteix en reduir el consum energètic, i això està relacionat directament amb la factura corresponent a l'edifici on s'aplica el projecte. Per tant, la inversió que suposa es veurà reflectida en un estalvi econòmic.

D'altra banda, el cost de personal difícilment pot ser més ajustat, ja que el desenvolupador és un estudiant que no percep beneficis econòmics. D'altra banda, com major sigui la inversió a nivell de sensors, més precís serà el sistema i això si que suposa una despesa en maquinari. A llarg termini s'espera que la despesa generi un estalvi superior.

5.3. Dimensió social

A any 2015 sembla ser que la situació econòmica comença a remuntar després d'uns quants anys en període de recessió. A nivell català, espanyol i europeu hi ha certa inestabilitat política, és època de reestructuració de les formes de govern. A nivell de carrer aquesta inestabilitat es veu reflectida en l'àmbit social, amb el distanciament entre classes altes i baixes.

L'objectiu del TFG és reduir el consum energètic mantenint la comoditat dels usuaris de l'edifici que l'utilitzi. Aquesta filosofia afavoreix a qualsevol persona, del nivell social que sigui, ja que li suposa un estalvi. A més, permet accions com apagar els llums d'una sala buida, alliberant de preocupacions al usuari i millorant així la seva qualitat de vida.

5.4. Dimensió ambiental

La taula 1 detalla els recursos que s'utilitzen a cada etapa del projecte. El consum és només elèctric i tots els components són de baix consum, seguint la filosofia del treball. A llarg termini s'espera que hi hagi un estalvi major a la despesa generada. A més, tot el material a excepció dels sensors específics per a aquest projecte és aprofitable per a qualsevol altre.

Tot el material que s'ha d'adquirir conté circuits elèctrics formats de materials que indirectament provenen de mines on es desconeix la situació real dels treballadors. Si bé aquest és un possible punt en contra l'ètica, el projecte compensa indirectament aquest factor, ja que redueix el desgast d'altres components elèctrics allargant-ne la seva vida útil.

Finalment, l'únic factor que no es controla és la generació de la energia que consumeix tota la infraestructura. Tant els components connectats a la xarxa com les bateries depenen del fabricant de cada element. Segons com es generin, la petjada ecològica serà millor o pitjor.

6. Disseny del sistema

6.1. Esquema

Per tal de crear la infraestructura del sistema, primer de tot ha estat necessari fer un disseny on poder veure i estudiar les diferents parts requerides i així després poder determinar quina era la solució més adient. En el següent esquema podem observar com es divideix tot el conjunt:

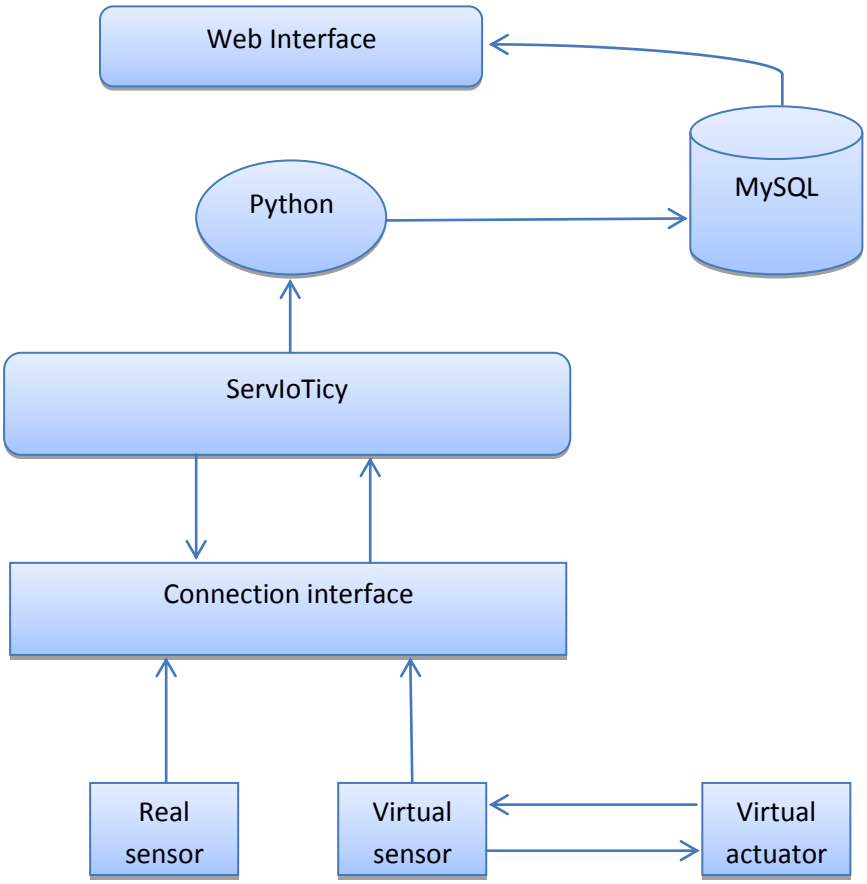


Figura 5 – Esquema del disseny

L'objectiu d'aquest sistema és el de captar dades de l'entorn d'un edifici i fer que, de manera transparent, un usuari extern o una aplicació puguin accedir-hi sense conèixer com està implementada la xarxa de sensors i actuadors real.

Aquest disseny disposa de diferents capes que permeten un aïllament del sistema respecte un usuari final. En el nivell més baix trobem els sensors i actuadors, ja siguin reals o virtuals. Aquest nivell és el que ens permet realment que el sistema tingui una utilitat, ja que és el que interactua directament amb l'entorn, sigui per captar-ne informació o per actuar-hi. Tots els sensors envien informació sobre la temperatura, humitat i lluminositat cada cert temps i ho fan amb un identificador únic que permet distingir-los.

En segon lloc ens trobem amb una capa intermèdia formada per una aplicació connectada directament amb els sensors (*connection interface*) i la plataforma encarregada de rebre les dades (*servIoTicy*). L'aplicació rep dades d'un sensor qualsevol sensor i els hi dóna el format necessari requerit per la plataforma. Aleshores, la plataforma emmagatzema les últimes dades de cada sensor i permet que aquestes puguin ser accedides externament per donar-ne ús.

Tenint les dades ja accessibles, el que volem és accedir-hi, i per això s'ha creat un programa *Python* que s'encarrega de llegir les dades noves cada vegada que s'ha produït una lectura en un dels sensors. El programa rep un missatge amb aquestes dades i les introdueix a una base de dades (*MySQL*). Les entrades d'aquesta base de dades són creades en el moment en que un nou sensor es connecta al sistema.

Un cop disposem de la informació en un lloc accessible, en aquest cas s'ha creat una interfície *web* on poder visualitzar-la fàcilment i així facilitar el control sobre el que esta passant en el nostre sistema.

Cal afegir que tant la base de dades com la interfície *web* són components per facilitar la visualització del sistema, però que en un treball futur no s'inclouran necessàriament en la infraestructura, sinó que les dades podran ser llegides directament de la plataforma *servIoTicy* i ser utilitzades segons les necessitats.

Finalment disposem d'una infraestructura amb una capa d'abstracció entre la xarxa de sensors i actuadors i l'usuari de les dades, que en farà l'ús que necessiti sense conèixer la implementació real.

6.2. Components

Seguint l'ordre de la explicació del disseny del sistema, a continuació es detalla cada un dels seus components, fent referència al seu funcionament, implementació i justificació del seu ús així com la comunicació existent entre ells.

Sensors reals

La base de la infraestructura són els sensors que capten dades. Actualment en aquest projecte es disposa d'alguns sensors reals. Concretament són capaços de captar informació de la temperatura, humitat i lluminositat del punt on es troben situats i comuniquen aquesta informació a través d'un mòdul sense fils de baix consum i curt abast.

El model escollit és el *AS-XM1000*[8] de l'empresa *Advanticsys,* un node sense fils que utilitza el protocol de comunicació 802.15.4, disposa d'una memòria *EEPROM* de 116 KB i porta incorporats els tres tipus de sensors esmentats, així com una connexió *USB* per connectar-se directament a un computador.

Els avantatges d'aquest mòdul concret són que disposa d'un disseny *hardware* basat en una plataforma pública i diverses facilitats en la configuració *software,* disposant de diferents exemples i suport que en permeten l'adaptació segons les necessitats desitjades.

Figura 6 - node AS-XM1000

El node esta basat en la plataforma original *TelosB* desenvolupada per la Universitat de Califòrnia, Berkeley. Incorpora un nou microprocessador amb 116 KB de memòria *EEPROM,* que millora la capacitat per instal·lar-hi la nostra aplicació respecte versions anteriors, i 8 KB de *RAM* així com les següents característiques:

- Plataforma *WSN IEEE* 802.15.4 compatible amb *TelosB,* per a la connexió sense fils.
- Microcontrolador *TI MSP430F2618, CC2420 RF*.
- Compatible amb *TinyOS* i *ContikiOS*.
- Sensors de temperatura, humitat i lluminositat.
- 2 polsadors multifunció.
- 3xLeds (blau, verd i vermell).
- Interfície *USB*.
- 2 bateries *AA* incloses.

Per informació més detallada sobre el node *AS-xm1000*, s'adjunta la seva especificació en el fitxer annex.

Aquests nodes funcionen amb piles, pel que no necessiten una presa de corrent elèctrica on connectar-se i d'aquesta manera podem col·locar-los en el punt de l'edifici que més ens convingui. El que si hem de tenir en compte és que l'antena de radio que tenen és de curt abast (uns deu metres) i necessiten una màquina propera que en capti les dades.

Cada cert temps, establert segons el programa que es carregui al node en concret, es fan lectures de temperatura, humitat i lluminositat i s'envien a la màquina receptora més propera. Aquestes dades s'envien mitjançant el protocol sense fils juntament amb un número identificador específic de cada node, no modificable. D'aquesta manera podem distingir quin és el que esta enviant informació.

El codi que executa cada node es troba en l'arxiu *SensorTestRadioC.nc* i esta implementat en llenguatge *C.* En iniciar-se el procés i un cop iniciada la comunicació de l'antena (funció per defecte *AMControl.start()*), s'executa la funció d'inici on hi establim el temps de lectura dels sensors (en mil·lisegons). En el següent codi d'exemple observem com s'ha establert un temps de 5000 ms (5 segons) entre una lectura i la següent:

```
   event void AMControl.startDone(error_t error){
      if (error == SUCCESS) {
         call Timer.startPeriodic(5000);
      }
      else {
         call AMControl.start();
      }
   }
```

Figura 7 - Codi d'inici del node

Si la comunicació amb l'antena falla es torna a intentar connectar (*else*) i si és satisfactòria aleshores estableix un *Timer* (*if*) per l'interval de temps entre lectures.

A partir d'aquest moment, cada 5 segons s'executarà automàticament una funció anomenada *Timer.fired().* Aquesta funció s'encarrega de fer tres crides a tres funcions específiques per a cada un dels sensors (*Temp.read()* per la temperatura, *Hum.read()* per la humitat i *Light.read()* per la lluminositat) i un cop rep les dades construeix el missatge que s'envia a través de l'antena de radio (contingut del *if*).

```
   event void Timer.fired(){
      call Temp.read();
      call Light.read();
      call Hum.read();
      call Batt.read();
      if (!busy) {
         SensorMsg* senspkt = (SensorMsg*) (call
Packet.getPayload(&pkt, sizeof(SensorMsg)));
         senspkt->nodeid = TOS_NODE_ID;
         senspkt->temp = TempVal;
         senspkt->hum = HumVal;
         senspkt->light = LightVal;
         senspkt->batt = BattVal;
         if (call AMSend.send(AM_BROADCAST_ADDR, &pkt,
sizeof(SensorMsg)) == SUCCESS) {
            busy = TRUE;
         }
      }
   }
```

Figura 8 - Codi d'execució automàtica segons el temps establert

Les tres funcions de lectura dels sensors simplement s'encarreguen de captar la dada llegida al moment i fer-ne la conversió adequada segons les unitats en les que volem obtenir-les realment (graus centígrads, percentatge d'humitat i nivell de lluminositat en *lux*). La seva implementació es pot veure en l'arxiu de codi adjunt anomenat *SensorTestRadioC.nc.*

Sensors i actuadors virtuals

En el desenvolupament d'aquest projecte, s'ha tingut en compte que l'objectiu era crear una infraestructura funcional que més endavant servirà per desenvolupar altres projectes en base la informació generada per aquest. D'aquesta manera, per simular el funcionament a més gran escala del sistema i sense haver d'invertir en la compra de sensors reals (amb el corresponent cost econòmic), s'ha afegit la possibilitat de connectar processos virtuals que emulin el comportament dels sensors reals així com el de possibles actuadors.

- Sensors virtuals. Estan formats per un procés independent que executa un codi *java*, el qual permet realitzar les mateixes funcions que un sensor real però amb dades aleatòries (segons certs paràmetres de configuració). Mitjançant una connexió a través d'un *socket,* envia el missatge amb les dades que simula haver captat de l'entorn cap a la interfície de connexió.

En iniciar-se el procés cal subministrar com a paràmetres d'entrada quin és el port de connexió del *socket* i quin fitxer de configuració li volem aplicar. En el següent exemple veiem com iniciar un procés, mitjançant una comanda en terminal *Linux*, que es connecta al port 9500 de la màquina en el que s'està a punt d'executar i es configura amb el fitxer *sensor_conf_v4.txt*. L'últim paràmetre (9502) és el port de connexió amb l'actuador virtual, que s'explica en el següent punt.

```
java client –port 9500 sensor_conf_v4.txt 9502
```

Figura 9 - Comanda execució sensor virtual

El fitxer de configuració conté informació sobre l'identificador del node virtual, un rang de possibles temperatures, humitats i intensitats de llum així com informació sobre la localització del node. A continuació un fitxer d'exemple:

```
#id
4
#temperature
-8,1
#humidity
65,75
#light1
555,600
#light2
550,580
#location
-
#room
D6121
```

Figura 10 - Fitxer de configuració sensor_conf_v4.txt

En l'exemple anterior tenim un identificador amb el número 4, un rang de temperatura que pot variar entre 8 graus sota zero i 1 grau positiu, humitat entre 65% i 75%, lluminositat entre 555 *lux* i 600 *lux* en el cas del sensor de llum #1 i de entre 550 *lux* i 580 *lux* per al sensor #2. A més en aquest cas no s'estableix la localització, que podria ser un nom d'un edifici i en canvi si es disposa del nom de l'habitació.

Un cop inicialitzat el procés, que executa el codi que segueix, cada 3 segons s'obté un valor de lectura dins els rangs establerts en la configuració (veure codi complet amb la funció que genera les dades anomenada *setValues,* adjunt en el fitxer *client.java*). La informació generada s'envia a través del *socket* amb el següent format: *id/temperature/humidity/light1/light2/location/room.*

```
public static void main(String argv[]) throws Exception {

        int port = 0;
        int portActuator = 0;
        String configFilePath = ""; //absolut path

        if (argv.length == 4) {
            if (argv[0].equals("-port")) {
              port = Integer.parseInt(argv[1]);
              configFilePath = argv[2];
              portActuator = Integer.parseInt(argv[3]);
            }
            else {
              System.exit(1);
            }
        }
        else {
            System.out.println("Usage: java client [-port
<PORT_ADDRESS>]");
            System.exit(1);
        }

        //check actuator socket
        vactuator = new VirtualActuator(portActuator);
        vactuator.start();

        while (true) {
            String sentence = "";
            sentence = setValues(configFilePath);
            Socket clientSocket = new Socket("localhost",
port);
            DataOutputStream outToServer = new
DataOutputStream(clientSocket.getOutputStream());
            outToServer.writeBytes(sentence + '\n');
            Thread.sleep(3000);
        }
    }
}
```

Figura 11 - Codi node sensor virtual

Les dues línies que segueixen el *//check actuator socket,* s'encarreguen de crear un fil d'execució paral·lel per a la comunicació amb els actuadors virtuals que es defineixen a continuació.

- Actuadors virtuals. De la mateixa manera que els sensors virtuals, estan formats per un procés independent que executa un codi *java.* A través d'una connexió amb un sensor virtual via *socket,* els actuadors virtuals permeten simular canvis en l'entorn d'afectació del sensor relacionat. Per exemple, un actuador pot ser capaç de pujar o baixar una persiana d'una habitació, la qual cosa implica un canvi de lluminositat que el sensor haurà de detectar.

En iniciar-se un procés d'actuador virtual, és necessari indicar quin és el port al qual s'ha de connectar per afectar a un sensor en concret. Aquest port ha de ser el mateix que el de l'últim paràmetre que s'indica en l'execució del procés del sensor virtual (port 9502 en l'exemple de la figura *C3*). En el següent exemple es mostra com executar un actuador virtual anomenat *actuatorBlind*, connectat via *socket* al port 9502.

```
java actuatorBlind 9502
```

Figura 12 - Comanda execució actuador virtual

```
public static void main(String argv[]) throws Exception {

        ...

         int previous = 0;
         Scanner scan = new Scanner(System.in);

         while (true) {
             System.out.println("Input '0' for blind down, '1'
for blind up");
             System.out.println("Current value: " +previous);
             int actual = scan.nextInt();
             previous = actual;
             String sentence = "";

             //0, decrement in '0' the light (L) value
             if (actual == 0) sentence = "L-/0";
             //1, increment in '200' the light (L) value
             else sentence = "L+/200";

             Socket clientSocket=new Socket("localhost", port);
             DataOutputStream outToServer = new
DataOutputStream(clientSocket.getOutputStream());
             outToServer.writeBytes(sentence + '\n');
         }
    }
```

Figura 13 - Codi procés actuador virtual

En la figura anterior podem veure què executa un actuador virtual. En el cas d'exemple, simulem un sistema que és capaç de pujar i baixar una persiana. Té dues posicions possibles, persiana abaixada (quan la variable *actual* val 0) i persiana oberta (quan la variable *actual* val 1). Suposem que en connectar-se aquest actuador, la persiana esta tancada i si introduïm un 1 per terminal durant l'execució del procés, aquest enviarà un missatge al sensor virtual a través del *socket* que els connecta amb el grau de lluminositat que s'ha variat a l'habitació (en introduir un 0 tornem a l'estat inicial).

El missatge a enviar està estructurat de la següent manera: *L+/200. L* indica que el valor a canviar és el de la lluminositat, *+* significa increment d'aquest valor i després de la barra el valor numèric indica la quantitat en que s'incrementa (o redueix si posem un -) el rang tant inferior com superior dels valors de lluminositat que ens dóna el sensor virtual corresponent. Si inicialment es dóna un valor de lluminositat d'entre 500 i 550, aleshores després d'executar l'actuador amb la persiana pujada, aquest rang passarà a ser d'entre 700 i 750. Si s'abaixa la persiana, aleshores tornarem a obtenir el rang original.

Aleshores, el procés del sensor virtual que s'està executant a la vegada i que disposa d'un fil d'execució independent que esta pendent de l'actuador connectat, rep el missatge amb els nous paràmetres que ha de llegir i donar com a lectura (en aquest cas canviar el rang de lluminositat).

La classe *VirtualActuator* de l'arxiu *client.java* conté la creació del fil d'execució paral·lel i la funció encarregada de rebre i interpretar el missatge. Les noves dades seran utilitzades mentre el actuador les modifiqui. Si aquest per contra no les ha canviat, es seguiran utilitzant les originals en el fitxer de configuració del sensor virtual.

```
public void run() {
        try {
            BufferedReader actData = new BufferedReader(new
InputStreamReader(actSocket.getInputStream()));
            String dataReceived = actData.readLine();
            String nodes = dataReceived;
            int j = 0;
            globals.type = "";
            globals.number = "";
            for (int i = 0; i < nodes.length(); i++) {
                if (nodes.charAt(i) != '/') {
                    if (j == 0) globals.type +=
nodes.charAt(i);
                    else if (j == 1) globals.number +=
nodes.charAt(i);
                }
                else ++j;
            }
            actSocket.close();
        }
        catch(IOException e) { e.getMessage(); }
    }
```

Figura 14 - Codi de recepció de les dades d'un actuador per part d'un sensor (virtuals)

Les variables globals *globals.type* i *globals.number* són les que indiquen el tipus i quantitat de paràmetre a modificar. Aquestes seran utilitzades per la funció de la classe *client* encarregada d'establir els valors de les variables de lluminositat que s'estan modificant amb l'actuador (entre altres), anomenada *setValues* i que es pot trobar en el fitxar adjunt *client.java*.

Interfície de connexió

La capa de connexió és l'encarregada de transmetre la informació rebuda des dels sensors cap a la plataforma que permetrà accedir públicament a aquestes dades. És constituïda per un programa *java* que capta totes les dades de tots els sensors que se li connectin, ja siguin reals o virtuals, donar-los el format necessari i enviar-les a la plataforma *servIoTicy.*

Aquest programa d'interfície ha de ser executat en una màquina propera físicament als sensors reals (els virtuals poden executar-se en aquesta mateixa màquina) ja que ha de disposar d'una antena que capti la informació dels sensors propers. Aquesta informació serà processada en primer terme localment i després serà enviada a través d'internet a la plataforma.

Les dades poden arribar de dues maneres, pel port sèrie de la màquina (per als sensors reals) o bé per un *socket* on hi hagi connectat un sensor virtual. Les dades seran tractades de diferent forma ja que el format de recepció és diferent per finalment obtenir el mateix tipus de dada que serà enviada a la plataforma en el format *json.*

Per iniciar el procés, executarem la següent comanda per terminal:

```
java demo –port 9500
```

Figura 15 - Comanda execució interfície de connexió, demo.java

```
public static void main(String[] args) throws Exception {

   ...

      PhoenixSource phoenix;

         if (port != null) {
           vreader = new VirtualReader(Integer.parseInt(port));
           vreader.start();
         }
         if (source != null) {
           phoenix =
BuildSource.makePhoenix(PrintStreamMessenger.err);
           MoteIF mif = new MoteIF(phoenix);
           demo serial = new demo(mif);
         }
}
```

Figura 16 – Codi funció principal demo.java

En la figura *C9* executem un nou procés que esta escoltant al port 9500 (*-port* ens indica que la connexió és d'un sensor virtual, si fos real hauria de ser *–comm* i li haurien d'indicar quin és el port a escoltar, per exemple un port serie) i en la *C10* observem el codi que executa la funció principal de la classe *demo.*

Si s'ha executat per connectar-hi un sensor virtual (*port != null*), aleshores s'iniciarà un nou fil d'execució dedicat al *socket* d'aquest sensor. Si en canvi s'espera un sensor real (*source != null*), es crearan certes instàncies de classes específiques del model de sensor real que estem utilitzant que permetran estar pendent dels missatges rebuts pel port sèrie.

Per als sensors virtuals aquest procés executarà la classe *VirtualReader* continguda en el propi *demo.java* i si és un sensor real executarà automàticament la funció *messageReceived* cada vegada que es rebin dades des del canal sèrie.

```
public void messageReceived(int to, Message message) {

    int source = message.getSerialPacket().get_header_src();
    //source = 3;
    uts = new Utils(source);
    uts.openDatafile("");   //Parameter. Empty string for real
sensor

    JSONObject obj = uts.parseMessage(message);

    if (obj != null) {
        uts.writeToFile(obj.toJSONString());
        uts.pushData("");   //Empty string for real sensor
    }
}
```

Figura 17 – Codi funció de recepció de missatges del port sèrie, arxiu demo.java

En rebre un nou missatge, s'executen diverses funcions contingudes en la classe *Utils* (inclosa en el mateix arxiu) encarregades d'obrir i crear si no existeix un fitxer local per guardar les dades en el format que requereix la plataforma *servIoTicy,* donar aquest format al missatge rebut, escriure'l i introduir-lo via internet a la plataforma. Totes aquestes funcions són, respectivament: *openDatafile*, *parseMessage*, *writeToFile* i *pushData*.

En el cas d'un missatge rebut via *socket*, les operacions a realitzar seran similars, tenint en compte que el format rebut és diferent i que la recepció la fem per un altre canal. Disposem d'un fil d'execució pendent de rebre dades en el port on s'ha connectat i, en fer-ho, el primer és interpretar el missatge per permetre tractar-lo de la mateixa manera que es fa amb un rebut des d'un sensor real.

Es reben dades en el format *id/temperature/humidity/light1/light2/location/room,* es separen individualment i es transformen en format *json* per ser tractades amb les mateixes funcions *openDatafile*, *writeToFile* i *pushData.*

- *openDataFile.* Obre un fitxer local anomenat *data_[id].json* (on *[id]* ens identifica que el fitxer correspon al un sensor virtual/real amb un nombre identificador). Si el fitxer no existeix, el crea localment seguint un model d'exemple (plantilla) i també crea un *service object* a la plataforma *servIoTicy* que ens retornarà un identificador. Aquest identificador es guarda en un altre fitxer local i és específic per utilitzar-lo amb la plataforma com a identificador del node sensor que està actualitzant dades.

- *enableMQTT.* S'executa només en connectar un nou node al sistema i habilita la subscripció a les actualitzacions que aquest fa regularment a la plataforma *servIoTicy.*

- *writeToFile.* Actualitza els valors de l'última lectura d'un node de sensors al fitxer local *data_[id].json.* Només emmagatzema la darrera actualització.

- *pushData.* Mitjançant el fitxers *data_[id].json,* el identificador rebut per part de la plataforma *servIoTicy* i una clau d'accés a aquesta plataforma, s'encarrega d'enviar-hi les noves dades llegides perquè hi siguin guardades.

A continuació s'explica el funcionament de la plataforma, així com el format necessari que han de tenir els fitxers *data_[id].json.*

Plataforma *servIoTicy*

Com s'ha anat introduint en apartats anteriors, aquesta plataforma esta connectada amb la interfície de connexió dels nodes sensors i permet oferir públicament a qualsevol usuari accedir a les dades captades pels nodes del sistema oferint una capa d'abstracció respecte la implementació de les capes inferiors. No és necessari doncs per al usuari final el coneixement del disseny i tecnologies aplicades al conjunt.

El nostre sistema disposa d'una clau d'accés a la plataforma per tal de poder emmagatzemar-hi les dades dels nostres sensors. En rebre una nova connexió, crea un identificador únic que serà el que permetrà identificar quin és el node que actualitza les dades en un moment donat.

El que ofereix aquest sistema és el següent: primer de tot es reben dades de qualsevol element (com són els nostres sensors) en un format que reconeix (*.json*). Quan oferim el fitxer amb el format i les dades que demana la plataforma, ja les tindrem disponibles per ser consultades externament. Poden ser rebudes per consultes d'usuaris externs o bé automàticament a través de subscripcions als identificadors dels diferents nodes de sensors que podem tenir.

En aquest projecte es disposa d'un programa que es subscriu a tots els nodes del sistema mitjançant el protocol *MQTT*[9]. És un protocol de connexió màquina a màquina *Internet of Things.* Està dissenyat per ser un sistema de missatgeria de publicacions/subscripcions extremadament lleuger.

En la següent imatge es mostra un esquema que resumeix el funcionament de la plataforma *servIoTicy.*

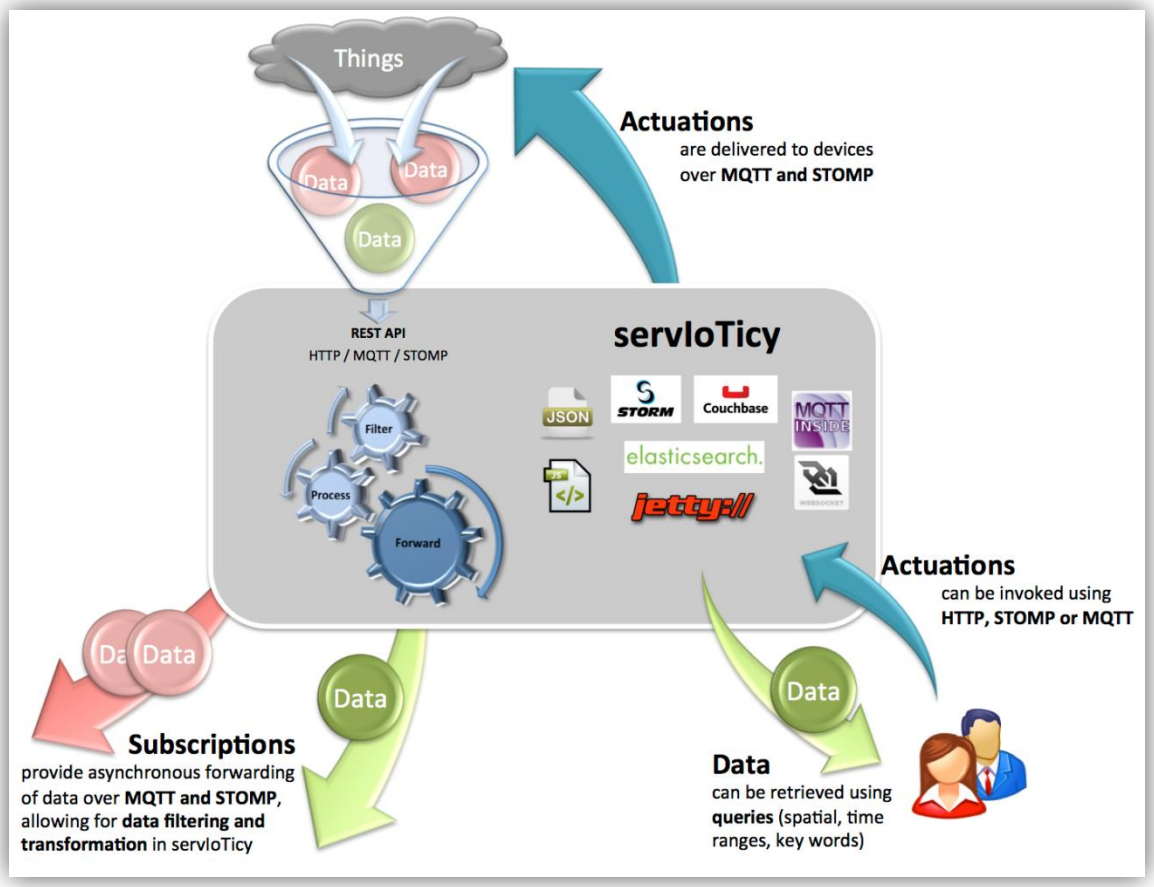


Figura 18 – Esquema plataforma servIoTicy

Els fitxers que la interfície de connexió envia a la plataforma són en format *.json,* com el d'aquest exemple:

```
{"lastUpdate":1433071427983,"channels":{"humidity":{"current-
value":69.0},"luminosity":{"current-
value":595.0},"temperature":{"current-value":-5.04}}}
```

Figura 19 – Contingut fitxer .json a enviar a la plataforma

El camp *lastUpdate* indica el moment en que s'ha generat la informació, en mil·lisegons. A partir de *channels* veiem tota la informació que volem enviar, com són els valors de la humitat, la lluminositat i la temperatura. Aquest tipus de fitxer que s'envia té aquesta estructura ja que en el moment de crear el *service object* (connexió de un nou node a la plataforma) s'ha fet seguint un model concret, el qual s'ha hagut de crear expressament amb els camps de dades desitjats.

En la pròxima figura veiem quina és la estructura d'un model. S'hi pot posar un nom i una descripció que identifiquin el fitxer amb un node del sistema i dintre del camp *streams* és on posem un *channels* amb els camps d'informació que es necessitin. Els camps restants són optatius per afegir informació extra.

```
{
    "name": "Virtual-Sensor-4",
    "description": "First virtual sensor",
    "streams": {
        "weather": {
            "channels": {
                "temperature": {
                    "type": "number",
                    "unit": "degrees"
                },
                "humidity": {
                    "type": "number",
                    "unit": "percentage"
                },
                "luminosity": {
                    "type": "number",
                    "unit": "light"
                }
            }
        }
    },
    "customFields": {},
    "actions": [],
    "properties": []
}
```

Figura 20 – Model format .json

Programa lector d'informació – *Python*

En aquest punt tot el sistema ja està implementat i operatiu, però perquè sigui realment funcional hem de verificar-ne el correcte funcionament. Per fer-ho, s'ha creat una aplicació en llenguatge *python* encarregada de rebre dades de la plataforma *servIoTicy* i actualitzar-les en una base de dades a la que accedirà una interfície *web.*

Per al seu funcionament necessitem tenir connexió a internet (per accedir allà on sigui la plataforma), la clau d'accés necessària per rebre les dades del sistema (*API KEY*) i un llistat dels identificadors públics dels nodes als quals ens volem subscriure.

En executar-se aquest programa primer de tot es llegeixen els identificadors corresponents als nodes de sensors dels que volem rebre informació i s'hi estableix una subscripció. Cada vegada que un node actualitzi la plataforma amb noves dades es rebrà automàticament un missatge amb aquestes.

Si és la primera vegada que es reben dades d'un identificador en concret, es crearà una nova entrada a la base de dades on hi anirem guardant la informació i si ja existeix s'actualitzarà amb l'últim valor obtingut.

El programa es comunica amb la plataforma mitjançant el protocol *MQTT,* que li permet crear una connexió, subscriure's als nodes desitjats i rebre informació automàticament cada vegada que aquesta sigui actualitzada. El fitxer *python* es pot consultar en l'arxiu adjunt anomenat *subs_reader.py.*

Base de dades – *MySQL*

Cada vegada que el programa *python* rep informació la envia a una base de dades. Aquesta base de dades (*MySQL*) està constituïda per una sola taula amb columnes corresponents al identificador de un node, la temperatura, humitat, llum (1 i 2, ja que els nodes disposen de dos sensors de llum), localització, habitació i posició. El camp de la localització indica quina és la situació de l'edifici on hi ha el sensor, l'habitació correspon a un nom que la identifiqui dintre l'edifici i la posició dintre l'habitació.

```
mysql> SELECT * FROM sensor_table;
+------------+-------+------+--------+--------+----------+-------+----------+
| id         | temp  | hum  | light1 | light2 | location | room  | position |
+------------+-------+------+--------+--------+----------+-------+----------+
| 0123456789 | 25    | 70   | 500    | -      | -        | D6121 | 1        |
| 1123456789 | 30    | 80   | 555    | -      | -        | D6114 | 1        |
| 2123456789 | 22    | 65   | 666    | -      | -        | D6121 | 8        |
| 0000000000 | 32    | 21   | 123    | -      | -        | D6114 | 9        |
| 7          | 22.58 | 70.  | 674.   | -      | -        | D6114 | 3        |
+------------+-------+------+--------+--------+----------+-------+----------+
```

Figura 21 – Taula MySQL

En crear una nova entrada a la taula, aquesta nomes disposa del camp identificador. La resta són buits (s'indica amb un '-'). Els valors de la localització, habitació i posició han de ser introduïts manualment en algun moment. La posició ha de ser un valor entre 1 i 9, com s'explicarà en el següent apartat sobre la interfície *web.*

```
...
#Insert new row in database, if does not exists
     cursor.execute("SELECT * FROM sensor_table WHERE id=7")
   rs = cursor.fetchall()
   if not rs:
      add_sensor = ("INSERT INTO sensor_table "
"(id,temp,hum,light1,light2,location,room,position)"
       "VALUES (%s,%s,%s,%s,%s,%s,%s,%s)")
      data_sensor = (ident,'-','-','-','-','-','-','-')
      cursor.execute(add_sensor,data_sensor)
...
```

Figura 22 – Inserció nova fila a la base de dades

En la figura anterior es mostra com es crea una nova entrada a la base de dades. Primer de tot es comprova, mitjançant un *SELECT*, si ja existeix una entrada amb el mateix identificador. Si no existeix (*if not rs*) aleshores s'insereix a la taula una nova fila (*INSERT INTO sensor_table*) amb tots els camps buits excepte el del identificador (*VALUES,* on els valors es defineixen en *data_sensor*). Es defineixen la comanda a executar i els valors a introduir i posteriorment s'executa la comanda (*cursor.execute(add_sensor,data_sensor))*.

Quan ja existeix l'entrada a la base de dades, s'ha d'introduir manualment els valors de la localització (opcional), habitació i la posició com es fa en els següents exemples:

```
mysql> UPDATE sensor_table SET position = '3' WHERE id = 7;
```

Figura 23 – Comanda introducció camp position

```
mysql> UPDATE sensor_table SET room = 'D6114' WHERE id = 7;
```

Figura 24 – Comanda introducció camp room

En el moment en que l'entrada ja existeixi a la base de dades i es rebi un nou missatge amb informació, el programa *python* executarà la següent comanda per tal d'actualitzar els valors del identificador *ident* i dels camp *temp, hum* i *light1.* A la següent figura es mostra el cas en que es modifica el valor de la humitat, en el camp *hum.* Per a la temperatura i la lluminositat només es modifica en el *SET* el nom de la columna corresponent.

```
cursor.execute("UPDATE sensor_table SET hum = %s WHERE id=%s",
               (value, ident))
```

Figura 25 – Comanda d'actualització del valor hum

Cada vegada doncs que un node de sensors llegeixi dades i les enviï a la plataforma mitjançant la interfície de connexió, el programa *python* s'encarregarà de rebre-les i actualitzar-les a la base de dades. Només guarda a la base de dades l'última actualització, ja que pel nostre propòsit (verificar el funcionament del sistema) és suficient.

Un cop tenim dades emmagatzemades a una base de dades hi accedirem mitjançant una interfície *web* que permetrà visualitzar-ne els valors com es descriu al següent apartat.

Interfície de visualització *web*

Per visualitzar els resultats del sistema implementat es disposa d'una interfície *web* que obté informació des de la base de dades anterior i la mostra gràficament sobre un mapa. Aquest mapa és un plànol de l'edifici on tenim els nodes de sensors instal·lats que ens permet veure'n la posició i quines són les lectures del moment. Per tant, permet obtenir informació sobre què està passant en cada una de les habitacions que es mostren en el plànol.

Aquesta interfície utilitza diferents llenguatges *web,* com *html, javascript* i *php.* Els dos primers són els que permeten representar i actualitzar sobre un navegador el resultat (com es mostra en la imatge posterior). El tercer realitza les consultes a la base de dades quan se li sol·licita des del navegador gràcies a la interacció amb l'usuari.

Un usuari que estigui visualitzant la interfície, ha de clicar amb el ratolí sobre l'habitació que desitgi per veure'n la informació corresponent. En clicar, s'executa una petició asíncrona a través del codi *php* i a la base de dades i quan es rep la informació es mostra per pantalla sense necessitat de refrescar el navegador, senzillament es modifiquen els camps afectats.

En rebre la informació sobre l'habitació seleccionada es pintaran de diferents colors punts que representen els diferents nodes instal·lats. Cada habitació es divideix en una matriu de 9 elements (per això en inserir la posició en la base de dades havia de ser un valor del 1 al 9) i cada un es representa amb un de nou colors diferents. Amb aquests colors podem relacionar el node amb un dels 9 quadres d'informació que es mostren sota el mapa, inicialment buits.

En clicar sobre el mapa també es rep la informació de les lectures dels sensors i aquesta serà introduïda en cada un dels quadres del color corresponent al node pintat a l'habitació seleccionada. Si una habitació té 3 nodes de sensors, només s'ompliran tres dels quadres d'informació (els dels colors que corresponguin) i la resta es deixaran en blanc.

Figura 26 – Interfície de visualització web

6.3. Funcionament pràctic

En aquest apartat es mostra quin és el pas a pas a seguir des de la posada en marxa del sistema fins a la visualització final, veient els passos i comunicacions entre els diferents components.

Inicialment hem d'arrencar la interfície de connexió. Aquest procés s'executa a una màquina que captarà dades de sensors propers. El connectarem al port 9500, del qual esperarà dades d'un sensor virtual. Aquest procés ha d'estar executant-se abans de connectar nodes al sistema.

```
sergi@sergi-HP-Pavilion-dv6-Notebook-PC: ~/TFG/part1/SensorTestRadio/src
sergi@sergi-HP-Pavilion-dv6-Notebook-PC:~/TFG/part1/SensorTestRadio/src$ java demo -port 9500
Server running
```

Figura 27 – Execució inicial interfície de connexió

El següent pas consisteix en connectar nodes al sistema, tant virtuals com reals. Si són reals, necessitarem una màquina propera amb una antena de ràdio capaç de rebre les dades a través del port sèrie. Els nodes *AS-XM1000* permeten realitzar aquesta funció, pel que en connectarem un via *USB* a un ordinador i en tindrem un altre a poca distància. En aquest mateix ordinador podem executar alguns processos de nodes de sensors virtuals.

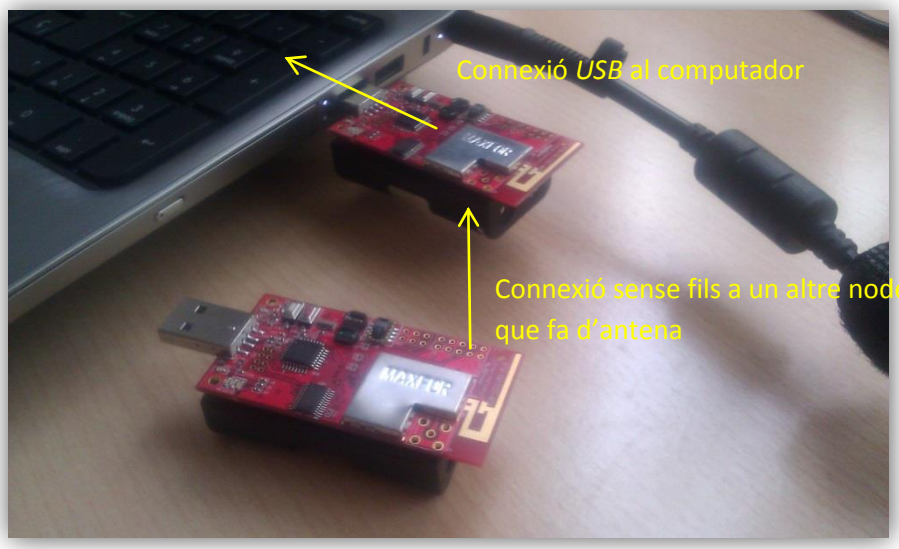


Figura 28 – Sensors reals connectats via USB i via antena ràdio

Si executem un nou procés de sensor virtual amb el identificador número 7, a través del port 9500, amb el fitxer de configuració *sensor_conf_v7.txt* i el port 9502 per connectar-hi un actuador virtual:

```
sergi@sergi-HP-Pavilion-dv6-Notebook-PC: ~/TFG/part1/SensorTestRadio/src
sergi@sergi-HP-Pavilion-dv6-Notebook-PC:~/TFG/part1/SensorTestRadio/src$ java client -port 9500 vConfigFiles/sensor_conf_v7.txt 9502
Client running
```

Figura 29 – Execució sensor virtual

En connectar-hi un node, la interfície de connexió comença a rebre missatges de cada lectura feta pels sensors. En la següent imatge podem observar com es crea i s'habiliten les subscripcions a un nou node i com es reben les dades llegides, després del comentari *File opened serviotic y/data/data_v7.json.* Es mostra un valor de temperatura (amb dos decimals), humitat i dues lectures de llum, tot i que després a la base de dades només es farà servir la primera.

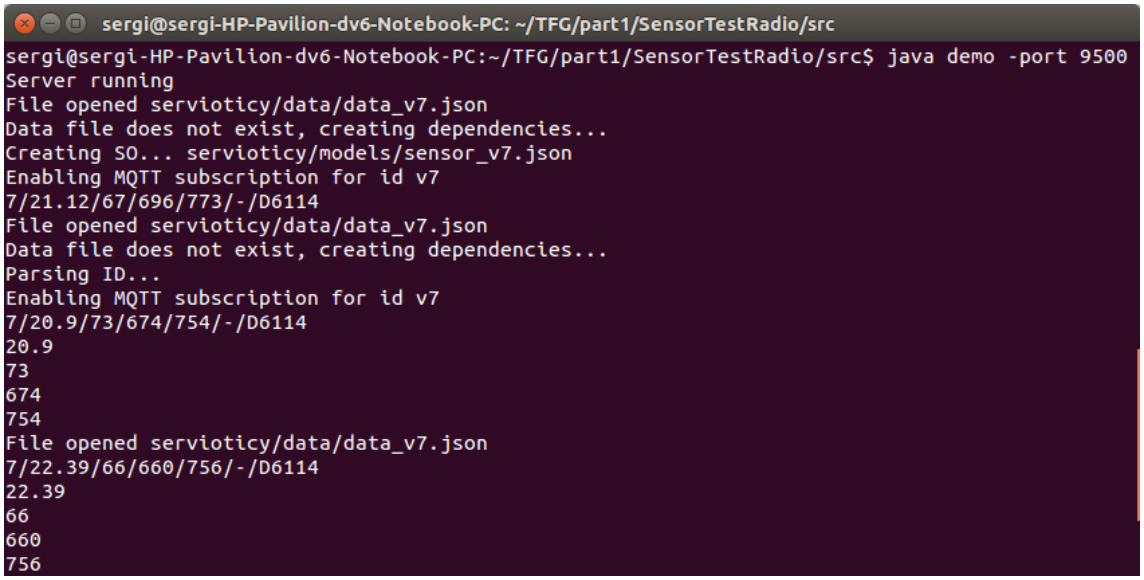


Figura 30 – Recepció dades a la interfície de connexió

A més, també iniciarem el procés d'actuador virtual que puja i baixa una persiana i es connectarà al port 9502.

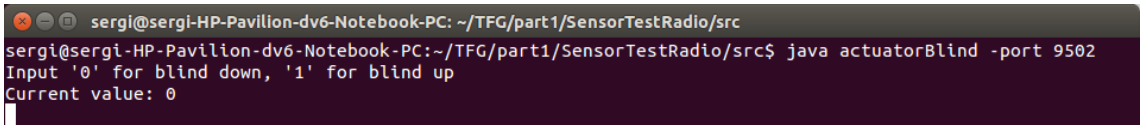


Figura 31 – Execució actuador virtual

Si pugem la persiana amb l'actuador virtual, el sensor virtual que està connectat augmentarà el seu rang de lluminositat 200 *lux.* Per fer-ho hem d'introduir un 1 i observarem el canvi des del terminal on s'executa la interfície de connexió.

En la imatge següent els valors de les lectures de lluminositat s'han incrementat en 200 *lux de* mitjana a causa de la pujada de persiana fet per un actuador.

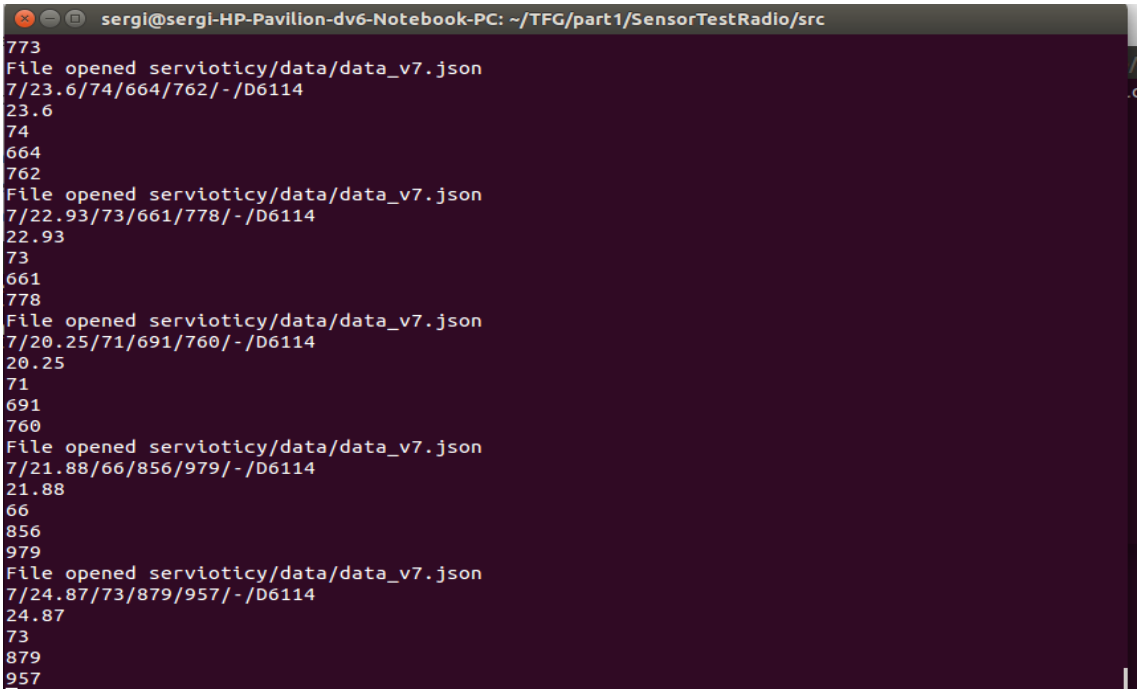


Figura 32 – Canvi de rang de dades causat per un actuador

Quan la interfície de connexió rep dades d'un nou node, genera un missatge de creació de *service object* a la plataforma *servIoTicy.* I hi va enviant el fitxer *data_v7.json* amb les noves lectures.

Amb la plataforma actualitzant lectures, en nostre programa *python* ens anirà actualitzant les dades a la base de dades, creant les noves entrades necessàries en les que haurem d'afegir manualment els camps *room* i *position.*

id	temp	hum	light1	light2	location	room	position
0123456789	25	70	500	-	-	D6121	1
1123456789	30	80	555	-	-	D6114	1
2123456789	22	65	666	-	-	D6121	8
0000000000	32	21	123	-	-	D6114	9
7	23.71	75.	654.	-	-	D6115	5

Figura 33 – Taula base de dades actualitzada amb nou node

En aquest punt ja es van rebent dades de cada lectura dels sensors i són visibles a través de la interfície *web.* Si en aquesta interfície es desplaça el ratolí, es veurà quina habitació s'està seleccionant i en clicar-hi s'actualitzarà la pàgina amb noves dades.

Figura 34 – Visualització final

En la darrera imatge veiem com, en finalitzar el procés, hem afegit un nou sensor a l'habitació *D6115* (posició número 5, color blanc) i que en ser l'únic de l'habitació, en clicar-hi només s'actualitza el quadre de color blanc. Cada vegada que cliquem es farà una consulta a la base de dades del sistema que estarà actualitzada amb l'última lectura i així es mostrarà en el quadre corresponent, el blanc en aquest cas.

7. Conclusions

El desenvolupament d'una infraestructura *IoT* enfocada als edificis és la base sobre la que desenvolupar una gran quantitat d'aplicacions que en facin ús. Gràcies al disseny d'aquesta infraestructura, es disposa d'un sistema en el que no és necessari conèixer la seva implementació per aprofitar-se'n.

L'objectiu d'aquest projecte era el de crear una base sobre la qual poder desenvolupar una aplicació que permeti millorar el consum energètic dels edificis. A més, es destacava la necessitat de fer-ho amb el mínim de consum per part dels diferents components (com els sensors). L'ús de tecnologies de baix consum com el protocol de comunicació 802.15.4 o el *MQTT,* aconsegueixen els objectius definits en aquest aspecte.

D'altre banda, el disseny realitzat conté una part a destacar com és l'ús d'una plataforma (*servIoTicy*) que permet l'accés via *Internet* des de qualsevol punt on es disposi d'una connexió i creant una capa d'abstracció de cara a l'usuari final de les dades. Això és beneficiós ja que aquest usuari no és depenent de la implementació del sistema i també crea un aïllament que serveix de protecció per a la xarxa de sensors. No se'n pot fer ús sense una clau específica.

Després de crear tota la base, s'ha dissenyat un sistema de verificació tal com es preveia inicialment, que permet comprovar com les dades es van actualitzant a mesura que es fan noves lectures per part dels sensors i que són accessibles externament, en el cas d'aquest treball des d'una aplicació *web.*

Finalment doncs, s'ha aconseguit arribar als objectius marcats inicialment amb la creació d'una infraestructura de baix consum i independent respecte una aplicació externa.

8. Treball futur

Aquest projecte forma part d'un de més gran que fa ús d'aquesta infraestructura per completar un sistema capaç de millorar el consum energètic dels edificis així com el confort dels seus usuaris.

El primer pas per continuar des d'on ha acabat aquest projecte consisteix en la creació d'un sistema d'aprenentatge automàtic (m*achine learnig*) que rebrà les dades de la plataforma *servIoTicy*. Aquest sistema haurà de ser capaç de preveure el comportament dels usuaris habituals de cada un dels espais d'un edifici per anticipar accions i corregir-ne en cas necessari.

Aquest aprenentatge ha de permetre actuar en els sistemes de consum energètic de l'edifici de manera que els seu ús garanteixi el confort minimitzant la despesa. Si per exemple en acabar una jornada laboral un empleat deixa un llum obert, el sistema ho detectarà i l'apagarà automàticament.

A més, per definir els paràmetres de confort que els usuaris de l'edifici volen, es necessària la interacció directa amb aquests. Per fer-ho es desenvoluparà una aplicació per a mòbils que permetrà a un usuari introduir quines són les seves pautes de comportament habituals o quin és el grau de lluminositat o temperatura que li resulta còmode. També li haurà de ser possible consultar sobre un plànol la situació del moment en l'espai que es troba.

Finalment, caldrà sumar les preferències dels usuaris amb les seves pautes de comportament apreses amb el pas del temps per establir el millor punt possible entre la reducció de despesa energètica i el confort de tots els usuaris.

9. Revisió del projecte

9.1. Tecnologies existents utilitzades

Després d'analitzar el sistema en la fase de disseny, s'ha decidit utilitzar la plataforma *ServIoTicy*. Aquesta plataforma ha estat desenvolupada inicialment pel *Barcelona Supercomputing Center – Centro Nacional de Supercomputación (BSC-CNS)* [6] a partir del projecte *COMPOSE* [7].

El que ens permet és disposar d'una capa d'abstracció entre tota la xarxa de sensors que capten i envien informació i la aplicació d'accés a aquesta informació. Sense cap cost en el projecte, disposem d'un sistema que ens dóna independència dels tipus de sensors que s'utilitzin (podent ser aquests diferents models entre ells o bé emulats).

L'aplicació que vulgui accedir a la informació, només haurà de subscriure's a cada un dels sensors desitjats segons l'identificador que la plataforma té per a cada un d'ells en particular. D'aquesta manera es permet accedir a una informació sense dependre de com està el sistema muntat realment. Només cal relacionar el identificador del sensor amb la seva situació a l'edifici per així representar-ne la seva ubicació real.

En aquest projecte es disposa d'un programa que actualitza una base de dades cada vegada que un sensor envia informació a la plataforma. Aquesta base de dades disposa de la informació més actualitzada de cada sensor i és on accedirà l'aplicació *web* que s'ha creat per representar el sistema i mostrar la informació.

9.2. Planificació final

Inicialment s'havia previst la implementació del sistema emulat i a continuació la del sistema real. Ja que durant la fase de desenvolupament s'han anat modificant alguna aspectes tècnics del sistema, ha estat necessari anar adaptant el sistema emulat, pel que en la fase de desenvolupament s'han implementat tots dos sistemes paral·lelament.

Si bé aquest aspecte no ha tingut afectació en la dedicació del projecte ni als seus costos associats, si que ho ha fet la necessitat de canviar parts del sistema. La utilització de la plataforma *servIoTicy* ha permès crear un sistema més genèric, que no depèn de la implementació o disseny de la xarxa de sensors però ha suposat un temps d'aprenentatge i adaptació no previst inicialment.

La plataforma s'ha decidit utilitzar-la quan la fase de desenvolupament ja s'havia iniciat, motiu pel qual el seu aprenentatge i introducció en el sistema ha allargat el temps d'execució del projecte.

Inicialment es pretenia finalitzar a dia 28 de maig de 2015. Ja que la data d'entrega i presentació reals són a partir de la tercera setmana del mes de juny, es disposen de dues setmanes extres per la finalització del treball. Es preveu finalitzar sense inconvenients ja que actualment s'està acabant la fase de desenvolupament i comprovant-te el seu correcte funcionament.

A nivell de costos, les hores de dedicació corresponents a les dues setmanes extres que es preveuen seran realitzades per l'estudiant, pel que no suposaran un cost econòmic extra.

9.3. Metodologia final

La metodologia que es preveia inicialment s'ha anat complint durant el projecte. Les etapes s'han executat satisfactòriament segons el procés previst i no ha estat necessari modificar-la per reconduir el treball. Les reunions setmanals amb el director del projecte han permès anar prenent les decisions necessàries de disseny i anàlisi de tot el sistema satisfactòriament.

9.4. Integració de coneixements

Pel desenvolupament d'aquest projecte s'han posat en comú diversos coneixements de diferents àmbits adquirits durant els estudis. La creació d'una infraestructura d'*Internet* de les coses requereix de conceptes de xarxes de computadors i de gestió de sistemes operatius. Part del programari ha de fer ús de la xarxa i per tant necessita accedir als recursos del computador adients.

A més, cal conèixer el funcionament dels sensors, que són dispositius específics de baix consum. És necessari saber l'estructura d'aquests components i quines són les seves limitacions així com els recursos que ens ofereixen. L'arquitectura d'aquests sistemes encastats condiciona el seu ús i capacitat.

El programari de la infraestructura de la xarxa de sensors ha estat realitzat amb el llenguatge de programació *Java,* tot i que la aplicació que utilitzen els sensors reals és implementada en llenguatge *C.*

En un últim apartat on la interacció amb l'usuari és més directe (interfície *web*), s'utilitzen sistemes de bases de dades i altres coneixements de desenvolupament *web*. En aquest cas es fa ús d'un sistema *Linux* amb una base de dades *MySQL* així com llenguatges de programació específics com són *PHP, HTML* i *Javascript.*

9.5. Lleis i regulacions

Les dades captades amb la xarxa de sensors són emmagatzemades a la plataforma *servIoTicy*. Seguint els seus termes d'ús i condicions, el projecte queda sota el compliment de les següents lleis, regulacions i llicències:

- El *software* de *servIoTicy* és distribuït sota la llicència *Apache*, versió 2.0. Permet el seu ús sota la llicència que l'usuari desitgi, tot i que altre programari pot ser distribuït sota una altra llicència, pel que cal verificar cada cas. Només es limita la seva distribució a llicències lliures i *open source* (codi accessible públicament).
- L'ús del programari i contingut així com les dades personals i altres continguts que estan disponibles a través de la plataforma apliquen les regulacions espanyoles que segueixen:
    - Llei orgànica espanyola 15/1999, de protecció de dades personals.
    - Reial decret legislatiu 1/1996, de 12 d'abril, que aprova el text de la llei espanyola de propietat intel·lectual.
- El *Barcelona Supercomputing Center – Centro Nacional de Supercomputación (BSC-CNS)* no participa en l'activitat realitzada en aquest projecte i que fa ús dels seus recursos.

ServIoTicy no es responsabilitza del contingut de les dades captades pels usuaris i emmagatzemades a la seva plataforma, pel que l'ús de les dades en aquest projecte no seran relacionades amb cap individu.

Per informació més detallada sobre els termes i condicions d'ús de la plataforma *servIoTicy*, consultar el seu lloc *web* [6].

10. Referències

10.1. Bibliografia

[1] Evans, Dave. (2011). *Internet de las cosas, cómo la próxima evolución de Internet lo cambia todo.* http://www.cisco.com/web/LA/soluciones/executive/assets/pdf/*Internet*-of-things-iot-ibsg.pdf

[2] Cisco®. (2015). *Internet de las cosas y la evolución de Internet*. http://www.cisco.com/web/ES/campaigns/*Internet*-de-las-cosas/index.html

[3] IBM®. (2015). *Smarter buildings*. http://www.ibm.com/smarterplanet/us/en/green_buildings/overview/

[4] Cidrera Lopez, Alain. (2013). *Plataforma smart building*. http://hdl.handle.net/2099.1/20781

[5] Kirschning, Ingrid. (1992). http://ict.udlap.mx/people/ingrid/ingrid/Tesis_EI/EI.html

[6] http://www.servioticy.com/

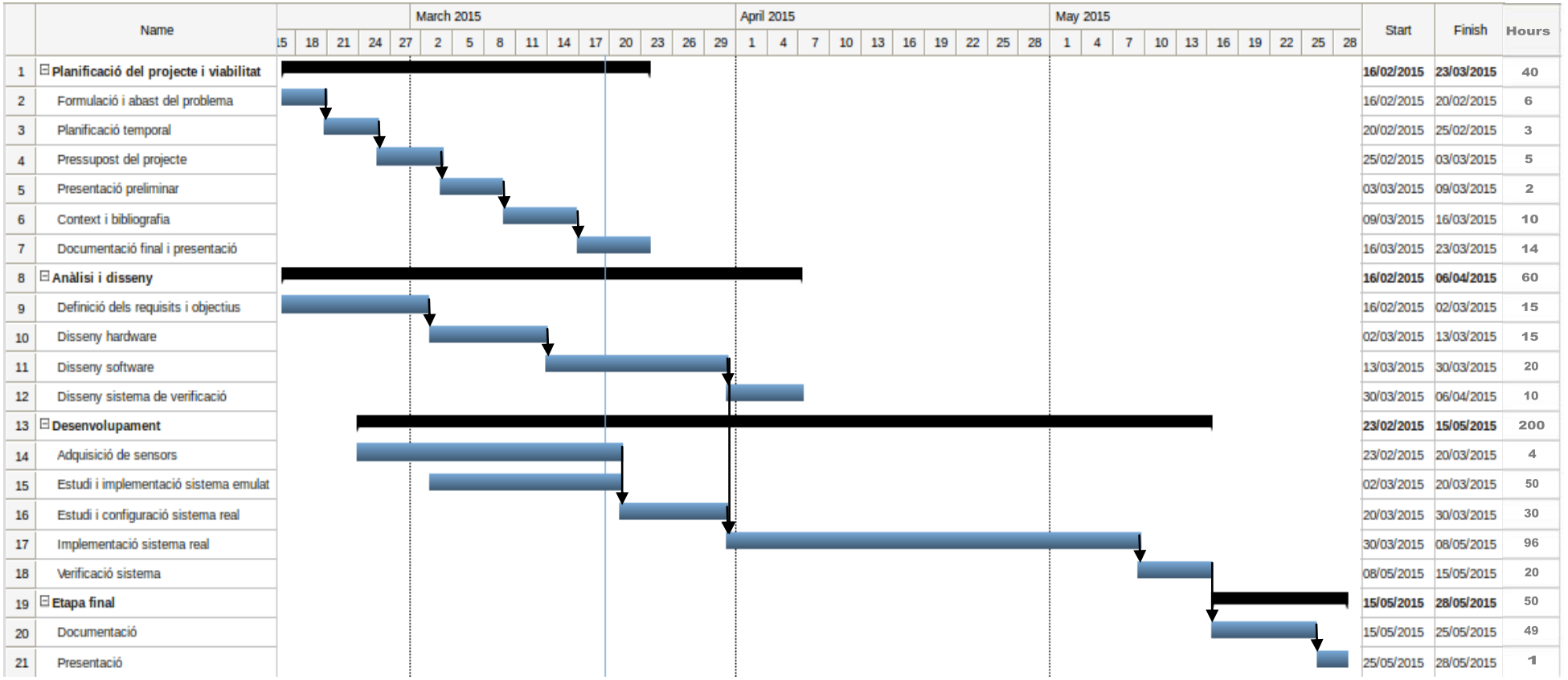
[7] http://www.compose-project.eu/

[8] http://www.advanticsys.com/shop/asxm1000-p-24.html?language=en

[9] http://mqtt.org/

11. Annex

11.1. Diagrama de *Gantt* inicial



	Name	Start	Finish	Hours
1	**Planificació del projecte i viabilitat**	**16/02/2015**	**23/03/2015**	40
2	Formulació i abast del problema	16/02/2015	20/02/2015	6
3	Planificació temporal	20/02/2015	25/02/2015	3
4	Pressupost del projecte	25/02/2015	03/03/2015	5
5	Presentació preliminar	03/03/2015	09/03/2015	2
6	Context i bibliografia	09/03/2015	16/03/2015	10
7	Documentació final i presentació	16/03/2015	23/03/2015	14
8	**Anàlisi i disseny**	**16/02/2015**	**06/04/2015**	60
9	Definició dels requisits i objectius	16/02/2015	02/03/2015	15
10	Disseny hardware	02/03/2015	13/03/2015	15
11	Disseny software	13/03/2015	30/03/2015	20
12	Disseny sistema de verificació	30/03/2015	06/04/2015	10
13	**Desenvolupament**	**23/02/2015**	**15/05/2015**	200
14	Adquisició de sensors	23/02/2015	20/03/2015	4
15	Estudi i implementació sistema emulat	02/03/2015	20/03/2015	50
16	Estudi i configuració sistema real	20/03/2015	30/03/2015	30
17	Implementació sistema real	30/03/2015	08/05/2015	96
18	Verificació sistema	08/05/2015	15/05/2015	20
19	**Etapa final**	**15/05/2015**	**28/05/2015**	50
20	Documentació	15/05/2015	25/05/2015	49
21	Presentació	25/05/2015	28/05/2015	1

11.2. Document control de gestió

Tasca	Estat	Comentaris
Planificació i viabilitat		
Formulació i abast del problema	Finalitzat	
Planificació temporal	Finalitzat	
Pressupost del projecte	Finalitzat	
Presentació preliminar	Finalitzat	
Context i bibliografia	Finalitzat	
Documentació final i presentació	Finalitzat	
Anàlisi i disseny		
Definició de requisits i objectius	Finalitzat	
Disseny *hardware*	Finalitzat	
Disseny *software*	Finalitzat	
Disseny sistema de verificació	Finalitzat	
Desenvolupament		
Adquisició sensors	Finalitzat	
Sistema emulat	Finalitzat	
Configuració sistema real	Finalitzat	
Implementació sistema real	Finalitzat	
Verificació del sistema	Finalitzat	
Etapa final		
Documentació	Finalitzat	
Presentació	Finalitzat	

11.3. Especificació *AS-XM1000*

ITEM	Specification	Description
Processor		
Processor Model	Texas Instruments® MSP430F2618	Texas Instruments® MSP430 family
Memory	116KB 8KB 1MB	Program flash Data RAM External Flash (ST® M25P80)
ADC	12bit resolution	8 channels
Interfaces	UART, SPI, I2C USB	Serial Interfaces External System Interface (FTI® FT232BM)
Radio		
RF Chip	Texas Instruments® CC2420	IEEE 802.15.4 2.4GHz Wireless Module
Frequency Band	2.4GHz ~ 2.485GHz	IEEE 802.15.4 compliant
Sensitivity	-95dBm typ	Receive Sensitivity
Transfer Rate	250Kbps	IEEE 802.15.4 compliant
RF Power	-25dBm ~ 0dBm	Software Configurable
Range	~120m(outdoor), 20~30m(indoor)	Longer ranges possible with optional SMA antenna attached
Current Draw	RX: 18.8mA TX: 17.4mA Sleep mode: 1uA	Lower RF Power Modes reduce consumption
RF Power Supply	2.1V ~ 3.6V	CC2420 Input Power
Antenna	Dipole Antenna / PCB Antenna	Additional SMA connector available for extra antenna
Sensors		
Light 1	Hamamatsu® S1087 Series	Visible Range (560 nm peak sensitivity wavelength)
Light 2	Hamamatsu® S1087 Series	Visible & Infrared Range (960 nm peak sensitivity wavelength)
Temperature & Humidity	Sensirion® SHT11	Temperature Range: -40 ~ 123.8 °C Temperature Resolution: : ± 0.01(typical) Temperature Accuracy: ± 0.4 °C (typical) Humidity Range: 0 ~ 100% RH Humidity Resolution: 0.05 (typical) Humidity Accuracy: ± 3 %RH (typical)
Electromechanical Characteristics		
Dimensions	81.90mm x 32.50mm x 6.55mm	Including USB connector
Weight	17.7g	Without batteries
Power	3V (2xAA Battery Holder Provided)	MICREL® MIC5207 Power Regulator